照会報告書作成プログラム

QMF (Windows 版) 入門

バージョン 7

SC88-8670-00
(英文原典：SC27-0723-00)

照会報告書作成プログラム

QMF (Windows 版) 入門

バージョン 7

SC88-8670-00
(英文原典：SC27-0723-00)

お願い

本書、ならびに本書で記述する製品をご使用になる前に、 155ページの『付録. 特記事項』を必ずお読みください。

本書は、DB2 ユニバーサル・データベース・サーバー (OS/390 版) (DB2 UDB (OS/390 版)) のバージョン 7 (プログラム番号 5675-DB2) のフィーチャーである照会報告書作成プログラム (Windows 版)、 DATABASE 2 サーバー (DB2 サーバー) (VM および VSE 版) のバージョン 7 (プログラム番号 5697-F42) のフィーチャーである照会報告書作成プログラム、AS/400 用照会報告書作成プログラム (Windows 版) (プログラム番号 5697-G24)、および DB2 ワークステーション・データベース用照会報告書作成プログラム (Windows 版) (プログラム番号 5697-G22)、さらに DB2 ウェアハウス・マネージャー (プログラム番号 5648-D35) および AS/400 用 DB2 ウェアハウス・マネージャー (プログラム番号 5697-G23) の照会報告書作成プログラム、さらに、改訂版で特に断りのない限り、以降のすべてのリリースおよび修正レベルにも適用されます。

本書は、旧版 GD88-7240-02 の大幅な改訂版です。本書における技術上の変更点は、該当変更個所の左端に縦線を付けて示しています。技術上の変更を含まない編集上の変更は特に示されていません。

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

http://www.ibm.com/jp/manuals/main/mail.html

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

http://www.ibm.com/jp/manuals/ の「ご注文について」をご覧ください。

(URL は、変更になる場合があります)

原　典： SC27–0723–00
Query Management Facility
Getting Started with QMF for Windows
Version 7

発　行： 日本アイ・ビー・エム株式会社

担　当： ナショナル・ランゲージ・サポート

第1刷 2000.7

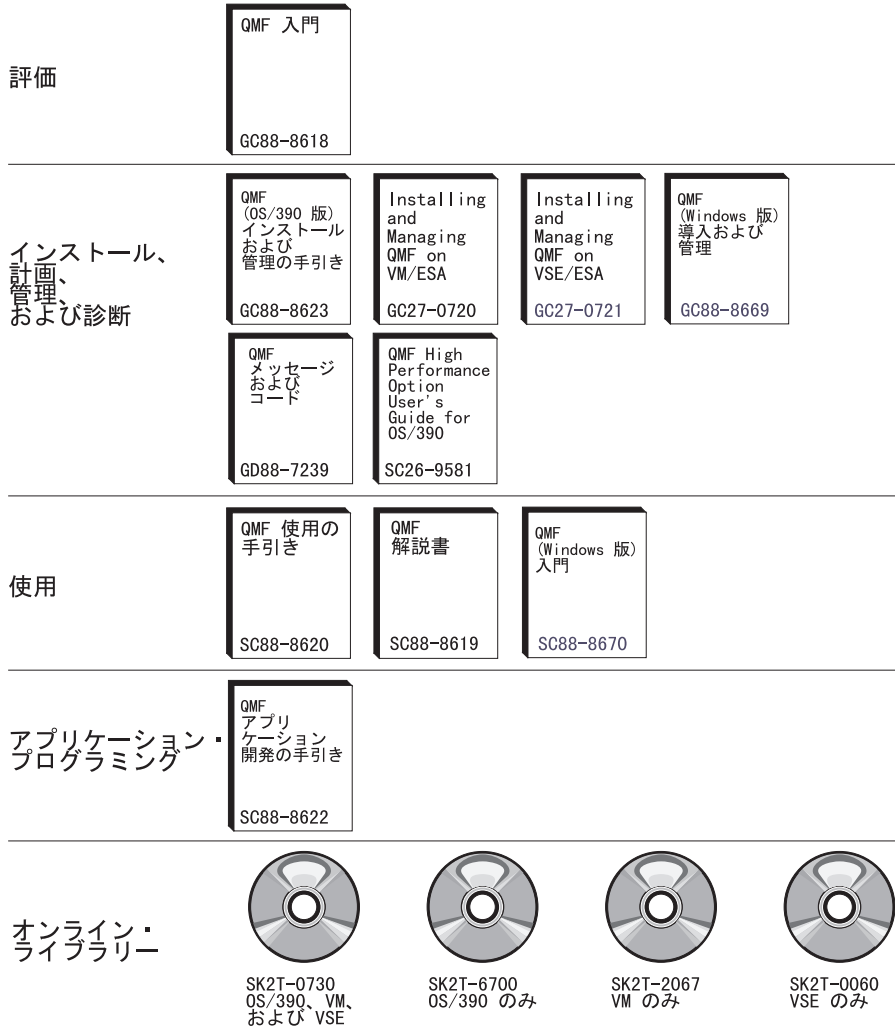
この文書では、平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、および平成角ゴシック体™W7を使用しています。この(書体*)は、(財) 日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注* 平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、
平成角ゴシック体™W5、平成角ゴシック体™W7

目次

QMF ライブラリー

資料は、 IBM 担当員を通じて注文してください。



第1章 概要

本章では、QMF (Windows 版) の概要を紹介し、QMF (Windows 版) の作動を開始するための基本的な作業のうちのいくつかを説明します。

データベース・サーバー

照会、書式、プロシージャーおよび表は、データベース・サーバーで実行され、保管されます。

DB2 ファミリーのデータベース

QMF (Windows 版) は、次のような広い範囲の DB2 データベースに接続することができます。

- DB2 UDB (OS/390 版)、 DB2 (OS/390 版)、および DB2 (MVS 版)
- DB2 サーバー (VSE および VM 版) および SQL/DS
- DB2 ユニバーサル・データベースおよび DB2 コモン・サーバー
- DB2 パラレル・エディション
- DataJoiner

QMF (Windows 版) のライセンスによって、QMF (Windows 版) のコピーが、どの DB2 ファミリー製品にインストールでき、接続できるかが決まります。

ユーザー名と技術名について

DB2 のバージョンおよびタイプが異なると、データベースを参照するのに、RDB 名を使用したり、ロケーション名を使用したり、あるいは他の技術名を使用したりします。

QMF (Windows 版) では、管理者が覚えやすい名前をデータベース名に割り当てることができます。たとえば、DB2P_01_PURCH の代わりに Purchasing Database (購買データベース) を使用します。

QMF (Windows 版) では、データベース・サーバーまたは DB2 データベースを単に「サーバー」と呼んでいます。

サーバー名の設定

データベースを照会できるためには、QMF (Windows 版) はまずデータベースが保管されている場所を知っている必要があります。

1. 「**ファイル**」メニューから、「**新規 SQL 照会**」を選択します。新規の SQL 照会文書が開かれます。
2. 「**照会**」メニューから、「**サーバーの設定**」を選択します。「サーバーの設定」 ダイアログ・ボックスが開かれます。

3. 使用可能なサーバーのリストから、照会を行うサーバーを選択し (「サーバーの選択」)、「**OK**」をクリックします。 次の QMF (Windows 版) セッションを開始するときに、 QMF (Windows 版) は自動的に同じサーバーに再接続します。

データベース・セキュリティー

サーバーに接続できるようになるには、まずユーザー ID とパスワードを指定しなければなりません。

ログオン

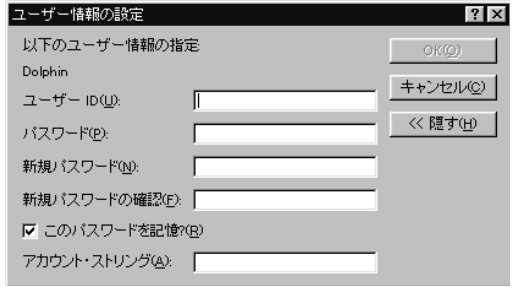
アクセスしようとするデータベース・サーバーに対して有効なユーザー ID とパスワードを指定しなければなりません。 データベース・サーバーのユーザー ID およびパスワードは、必ずしもローカルまたはネットワークのユーザー ID およびパスワードと同じであるとは限りません。

Windows を実行している場合は、QMF (Windows 版) の複数のセッションにわたって、サーバー・パスワードを記憶しているいうオプションがあります。現在 Windows にログオンしている場合、「ユーザー情報の設定」ダイアログ・ボックスは、「**このパスワードを記憶?**」というラベルが付いた追加のチェック・ボックスを表示します。このチェック・ボックスを選択すると、そのサーバー用に入力したパスワードが、Windows パスワード・リストに保管されま

す。パスワードを記憶させたユーザーが Windows にログオンしている場合には、QMF (Windows 版) は、常にそのパスワードを自動的に検索して、ユーザーにパスワードを入力するようプロンプトを出さないようにすることができます。 QMF (Windows 版) を実行するときに、パスワードを記憶させたユーザーがログオンしていない場合、あるいは、別のユーザーとしてログオンしている場合、QMF (Windows 版) はユーザー ID とパスワードを入力するようにプロンプトを出します。

注: パスワードを保管するように決めた場合、そのユーザーの Windows のアカウントにログオンできる人はだれでも、そのユーザー (サーバー) のユーザー ID とパスワードを使用して、そのユーザーのデータベース・サーバーにアクセスすることができます。

1. 「**照会**」メニューから、「**ユーザー情報の設定**」を選択します。すると「ユーザー情報の設定」ダイアログ・ボックスが開きます。



2. ユーザー ID とパスワードを該当のフィールドに入力します。

   注: ユーザー ID とパスワードは、大文字小文字の区別があります。たとえば、ユーザー ID またはパスワードが英大文字である場合には、それを英大文字で入力することが必要です、データベース・サーバーのタイプによっては、ユーザー ID とパスワードを、大文字小文字の区別があるものとして扱うものと、大文字小文字の区別がないものとして扱うものがあります。

3. ユーザー ID とパスワードを保管する場合には、「**このパスワードを記憶?**」にチェックを付けます。
4. 「**OK**」をクリックします。すると QMF (Windows 版) は、サーバーをアクセスする準備のため、この情報を保管します。

パスワードの訂正

正しくないパスワードを入力した場合、「ユーザー情報の設定」ダイアログ・ボックスを再度開いて、エラーを訂正することができます。

1. 「**照会**」メニューから、「**ユーザー情報の設定**」をクリックします。 すると「ユーザー情報の設定」ダイアログ・ボックスが開きます。
2. パスワードを再入力して、「**OK**」をクリックします。 パスワードが訂正されます。

パスワードの変更

データベース・サーバーで、QMF (Windows 版) から、パスワードを変更することができます。この機能は、現在は、DB2 (OS/390 版) バージョン 5 以降でのみサポートされています。

1. 「**照会**」メニューから、「**ユーザー情報の設定**」を選択します。すると「ユーザー情報の設定」ダイアログ・ボックスが開きます。
2. 「**変更**」をクリックします。「**新規パスワード**」および「**新規パスワードの確認**」のフィールドが表示されます。
3. ユーザーの新規パスワードを、「**新規パスワード**」と「**新規パスワードの確認**」のフィールドに入力して、「**OK**」をクリックします。これで、ユーザーのデータベース・サーバー・パスワードが変更されます。

アカウント・ストリングの指定

データベース・サーバーは、アカウント・ストリングを使用して、システムの使用状況を追跡します。データベース管理者に、このシステムがアカウント・ストリングを使用しているかどうかを確かめてください。

1. 「**照会**」ニューから、「**ユーザー情報の設定**」を選択します。すると「ユーザー情報の設定」ダイアログ・ボックスが開きます。
2. 「**アカウント・ストリング**」フィールドで、使用するアカウント・ストリングを入力し、「**OK**」をクリックします。すると QMF (Windows 版) は、サーバーをアクセスする準備のために、この情報を保管します。

管理

QMF (Windows 版) 管理プログラムは、常にバックグラウンドで実行し、データベースおよびシステム・リソースの使用状況を監視しています。また、管理プログラムは実行できる照会のタイプおよびサイズも制限します。

リソース限界の表示

「**表示**」メニューから、「**リソースの限界**」を選択します。「リソース限界」ダイアログ・ボックスが開きます。「リソース限界」ダイアログ・ボックス内

の情報はすべて読取専用です。システム管理者がこれらの限界を設定します。

有効な限界および制御のタイプは次のとおりです。

- タイムアウト
- 限界
- SQL 動詞
- オプション
- データの保管
- バインディング
- オブジェクトの追跡

限界行数の設定

該当の照会で検索する最大 (限界) 行数を指定することができます。この限界に到達すると、QMF (Windows 版) はその照会を取り消します。リソース限界グループに指定されている最大の許可限界行数は、このパラメーターより優先されます。

このフィールドに限界を設定しない場合には、0 を入力します。

この限界を超えて QMF (Windows 版) がすでに検索している行は、表示するために保持され、使用可能になっています。

1. 「**照会**」メニューから、「**限界行数の設定**」をクリックします。「限界行数の設定」ダイアログ・ボックスが開かれます。

2. 照会で戻す最大の行数を入力し、「**OK**」をクリックします。限界行数は、次にこの照会を実行したときに適用されます。

ツールバーの変更

ツールバーを変更して、参照したいボタンだけを表示することができます。

ツールバーへのボタンの追加

既存の QMF (Windows 版) のツールバーにボタンを追加するオプションがあります。これらのボタンは、すべてのユーザーが必要とするものではないが、ツールバーに組み込んで使用できる機能を表しています。

1. ツールバーを囲むグレーの区域をダブルクリックします。「ツールバーの変更」ダイアログ・ボックスが開かれます。

2. 「**利用できるボタン**」列から追加するボタンを選択して、「**追加**」をクリックします。 そのボタンがツールバーに追加されます。
3. ボタンの追加が完了したら、「**閉じる**」をクリックします。ダイアログ・ボックスが閉じて、新規のボタンがツールバーに追加されます。

ツールバー上のボタンの移動

QMF (Windows 版) のツールバー上のボタンを並べ替えるオプションがあります。

1. ツールバーを囲むグレーの区域をダブルクリックします。「ツールバーの変更」ダイアログ・ボックスが開かれます。
2. 「**利用できるボタン**」列から、移動したいボタンを選択します。

3. 「**上へ**」および「**下へ**」ボタンを使用して、ツールバー内でボタンを移動します。
4. ボタンの移動が完了したら、「**閉じる**」をクリックします。ダイアログ・ボックスが閉じて、ボタンがツールバー上の新規の位置に表示されます。

ツールバーからのボタンの削除

QMF (Windows 版) のツールバーからボタンを削除するオプションがあります。

1. ツールバーを囲むグレーの区域をダブルクリックします。「ツールバーの変更」ダイアログ・ボックスが開かれます。
2. 「**利用できるボタン**」列から除去するボタンを選択して、「**除去**」をクリックします。そのボタンがツールバーから除去されます。
3. ボタンの除去が完了したら、「**閉じる**」をクリックします。ダイアログ・ボックスが閉じて、そのボタンがツールバーから除去されます。

第2章 SQL 照会の作業

SQL (構造化照会言語) は、ユーザーとデータベースとの間の最も基本的なインターフェースです。照会は SQL で書かれ、データベースで処理されます。ユーザーは、QMF (Windows 版) の照会を SQL で書くことができますが、「ポイント・アンド・クリック (マウスでポイントしてクリックする)」方式でも照会を作成することができます。

SQL 照会

構造化プログラミング言語の照会を行うためには、SQL のコマンドとその構文を知っている必要があります。 SQL について詳しくない場合は、指示照会の方法を使ってみてください。

新規の SQL 照会の作成

ツールバーの「**新規 SQL 照会**」ボタンをクリックします。

新規の照会文書が開かれます。

データベース・サーバーでの SQL 照会の実行

1. 新規の照会文書を開き照会を入力するか、または既存の照会ファイルを開くか、またはデータベースから照会を開きます。
2. ツールバーの「**照会実行**」ボタンをクリックします。

照会が実行され、結果が表示されます。

結果の表示と SQL 表示の間の切り替え

照会の結果または SQL ステートメント自身のいずれかを見ることができます。

実行した照会の「SQL 表示」から、ツールバーの「**結果の表示**」ボタンをクリックします。

照会の結果が表示されます。

- あるいは -

照会の「結果の表示」から、「**SQL 表示**」ボタンをクリックします。

SQL ステートメントが表示されます。

フォントに関する作業

照会を表示するために使用されるフォントを変更することができます。選択できるフォントは、使用しているコンピューターにインストールされているものによって異なります。フォントの追加について詳しくは、使用しているオペレーティング・システムのヘルプ機能を参照してください。

注: 新規の照会表示フォントを選択した後で照会を保管すると、その照会は、常に新規フォントを使用して表示されます。

照会表示フォントの選択

1. 「SQL 表示」から、「**照会**」メニューの「**フォントの設定**」をクリックします。 「フォント」ダイアログ・ボックスが開かれます。
2. 照会のテキストを表示するフォントを選択して、「**OK**」をクリックします。照会が新規のフォントで再表示されます。

   注: 「**デフォルト設定**」をクリックすると、選択したフォントがすべての新規照会のデフォルト・フォントとして使用されます。

複数の照会

複数の照会文書を同時に開いておくことができます。 また、複数の照会を同時に実行することもできます。 この機能を使用して、複数の報告書が生成され、またある照会から別の照会へ、SQL テキストを切り取って貼り付けることができます。

同時に複数の照会の表示

1. 少なくとも 2 つの照会文書を開きます。
2. 「**ウィンドウ**」メニューから、次のいずれかのコマンドを選択します。

コマンド	結果
重ねて表示	選択すると、複数の照会が少しずつずれた形で重なって表示されます。
横のタイル表示	選択すると、照会ウィンドウが縦に積み上げられて表示されます。
縦のタイル表示	選択すると、照会ウィンドウが横に並べられて表示されます。

照会ウィンドウは、選択したオプションに従って配置されます。

照会のドロー

新規 SQL 照会文書の作成を支援するために、「照会のドロー」コマンドを使用します。必要とする 1 つまたは複数の表名および SQL ステートメントのタイプを指定すると、QMF (Windows 版) は、表内の列の名前およびデータ・タイプを参照する SQL ステートメントを自動的に作成します。

新規の SQL 照会の作成

1. 「**ファイル**」メニューから、「**照会のドロー**」をクリックします。「照会のドロー」ダイアログ・ボックスが開かれます。

2. 作成する照会のタイプを選択します。

照会タイプ	結果
選択	1 つまたは複数の表から行を検索します。
更新	表内にある情報を変更します。
挿入	表に新規の行を追加します。

3. 照会する表の所有者および名前を入力します。

   注: マッチング表のリストから表名を選択する際に、パターンを使用することができます。

   - 任意の文字を含む任意の長さのストリングのマッチングを行うためには、パーセント文字 (%) を使用します。 たとえば、名前が文字 A で始まる表をすべてリストするためには A% と入力します。
   - 単一文字のマッチングを行うためには、下線文字 (_) を使用します。 たとえば、 2 番目の位置に文字 A が入った所有者の表をすべてリストするためには _A% と入力します。

   パターンの入力後、「**リストから追加**」をクリックし、結果のリストから表を選択します。

4. その表の固有 ID を入力します。
5. 「**追加**」をクリックします。表が照会に追加されます。

6. 照会する表 (1 つまたは複数) を追加して、「**OK**」をクリックします。 選択した表の SQL 照会が作成され、表示されます。

SQL 照会での置換変数

置換変数を使用して、照会を実行するごとに異なる値を提供すると、同じ照会を使用して異なる情報を検索することができます。異なるデータの組を検索するために照会を書き直す必要はありません。その場合は、照会を実行する際に、照会の置換変数に異なる値を提供してください。

置換変数とは、照会に組み込まれるテキストのことです。置換変数は、アンパーサンド文字 (&) で始める必要があり、18 文字までの英字、数字、または ^ ! $ ~ { } ? @ # % ¥ _ の特殊文字の 1 つを持つことができます。たとえば、次に示すものは有効な置換変数です。

```
&VARIABLE1
&DEPARTMENT_NUMBER
```

置換変数は照会内のどこに現れてもよく、変数の値は照会に書き込めるものであれば何でもかまいません (ただしコメントは除きます)。 たとえば、列名、検索条件、副照会、または任意の特定の値の代わりに置換変数を使用できます。

置換変数を使用する SQL 照会の実行

1. 新規照会文書を開いて、次の SQL ステートメントを入力します。

   ```
   SELECT * FROM Q.STAFF WHERE DEPT >= &MIN_DEPT
   ```

2. 照会を実行します。「置換変数値の入力」ダイアログ・ボックスが開かれます。

3. 「**値**」フィールドに 50 の値を入力して、「**OK**」をクリックします。照会が実行され、照会結果が表示されます。

SELECT および FROM 文節内の値を置換して、置換変数を試してください。異なる入力によって、照会が戻す結果を確認してください。

SQL 照会を保管および開く

照会は、PC、ファイル・サーバー、またはデータベース・サーバーに保管することができます。

ファイルへの SQL 照会を保管

1. 「照会を開く」から、ツールバーの「**保管**」ボタンをクリックします。

照会が以前に保管されている場合も、その照会が再度保管されます。指示照会が以前に保管されていない場合は、「別名保管」ダイアログ・ボックスが開かれます。

2. 照会を保管するファイルの名前を入力して、「**OK**」をクリックします。 照会が保管されます。

保管した SQL 照会ファイルを開く

1. ツールバーの「**開く**」ボタンをクリックします。

「開く」ダイアログ・ボックスが開かれます。

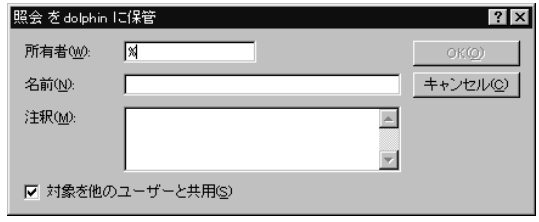
2. 開くファイルを選択して、「**OK**」をクリックします。選択した照会が、新規照会文書内に開かれます。

データベース・サーバーで SQL 照会を保管

サーバーに保管されている照会は、他のユーザーからアクセスできるようにすることができます。照会を他のユーザーと共用させたい場合は、照会をデータベース・サーバーに保管してください。

1. 「照会を開く」から、ツールバーの「**サーバーに保管**」ボタンをクリックします。

「照会の保管」ダイアログ・ボックスが開かれます。



2. 所有者、名前を入力し、保管されている照会を他のユーザーと共用するのかどうかを選択し、「**OK**」をクリックします。 照会がサーバーに保管されます。

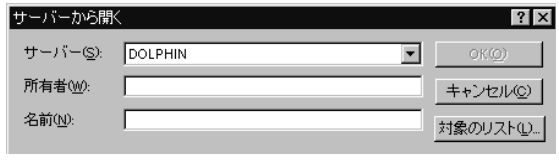
   この名前の照会がすでに存在している場合は、前から存在している照会を上書きするようにプロンプトで指示されます。

データベース・サーバーでの保管した SQL 照会を開く

ユーザーは、データベース・サーバーに保管されている照会を開くことができます。

1. ツールバーの「**サーバーから開く**」ボタンをクリックします。

「サーバーから開く」ダイアログ・ボックスが開かれます。



2. サーバー、所有者、および名前を入力し、「**OK**」をクリックします。 SQL 照会が開かれます。

SQL 照会の印刷

SQL 照会のプリビューと印刷を行うことができます。

照会のプリビュー

1. 照会を開いて「SQL 表示」を起動します。 SQL ステートメントが表示されます。
2. 「**ファイル**」メニューから、「**ページの設定**」をクリックします。 「ページの設定」ダイアログ・ボックスが開かれます。
3. ページのレイアウトに、必要な変更を行い、「**OK**」をクリックします。
4. ツールバーの「**印刷プリビュー**」ボタンをクリックします。

印刷する照会のプリビューが現れます。

SQL 照会の印刷

1. 照会を開いて「SQL 表示」を起動します。 SQL ステートメントが表示されます。
2. 「**ファイル**」メニューから、「**ページの設定**」をクリックします。 「ページの設定」ダイアログ・ボックスが開かれます。
3. ページのレイアウトに、必要な変更を行い、「**OK**」をクリックします。
4. ツールバーの「**印刷**」ボタンをクリックします。

照会が印刷されます。

第3章 指示照会に関する作業

指示照会は、メニューおよびリストからオプションを選択して照会を作成する、簡単な方法です。指示照会は、作成してしまうと、保管することも、SQL照会に変換することもできます。

単純な照会の作成

指示照会インターフェースを使用すれば、単純な照会を作成することができます。

新規指示照会を開く

- 「**ファイル**」メニューから、「**新規指示照会**」をクリックします。新規の「指示照会」文書が開かれます。

指示照会アクション・ボタン

照会アクション・ボタンを使用して、指示照会を編集します。 ボタンのセットが、制御するセクションの上に表示されます。

指示照会 アクション・ボタン	外観	結果
追加		クリックすると、指示照会に項目が追加されます。

編集		クリックして、照会内で強調表示されている項目を編集します。
削除		クリックすると、選択した項目が削除されます。
「上へ移動」および「下へ移動」		クリックすると、指示照会内で選択した項目が上下に移動します。

指示照会への表の追加

1. 「指示照会」文書の「表」セクションで、「**追加**」ボタンをクリックします。

   「表」ダイアログ・ボックスが開かれます。

2. 追加する表の所有者および表名を入力して、「**追加**」をクリックします。表が照会に追加されます。

   注: マッチングするオブジェクトのリストからオブジェクトを選択する際に、パターンを使用することができます。

   - 任意の文字を含む任意の長さのストリングのマッチングを行うためには、パーセント文字 (%) を使用します。 たとえば、名前が文字 A で始まる表をすべてリストするためには A% と入力します。
   - 単一文字のマッチングを行うためには、下線文字 (_) を使用します。たとえば、2 番目の位置に文字 A が入っている所有者の表をすべてリストするためには _A% と入力します。

   パターンを入力した後、「**リストから追加**」をクリックし、結果のリストから表を選択します。

3. 照会に追加の表条件を追加して、「**閉じる**」をクリックします。新規の表をリストした指示照会文書が表示されます。

指示照会の実行

指示照会は、SQL 照会を実行するのと同じ方法で実行します。ツールバーの「照会実行」ボタンをクリックします。

指示照会が実行されます。

複雑な照会の作成

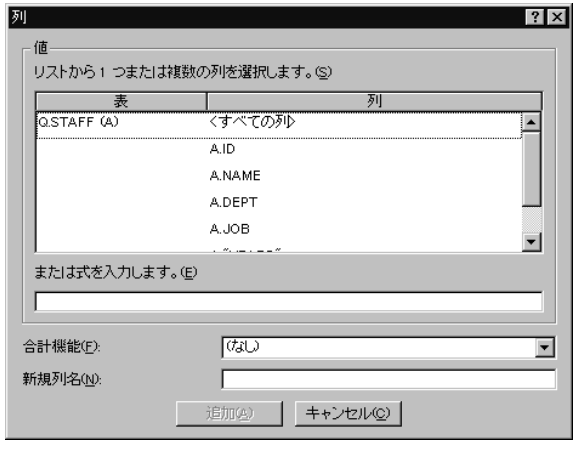
指示照会インターフェースを使用して、さらに複雑な照会も作成することができます。

指示照会への列の追加

1. 「指示照会」文書の「列」セクションで、 「**追加**」ボタンをクリックします。

   「列」ダイアログ・ボックスが開かれます。

2. 追加する列を選択して、「**追加**」をクリックします。列が指示照会に追加されます。
3. 照会に、追加の列を追加して、「**閉じる**」をクリックします。新規の列をリストした「指示照会」文書が表示されます。

注: 「機能」フィールドで合計機能を選択すると、該当の列に合計機能を適用することができます。使用可能な合計機能としては、AVERAGE、COUNT、MAXIMUM、MINIMUM、および SUM があります。

注: 「**新規列名**」フィールドに新規の列名を入力して、照会の中の列の名前を変更することができます。

ソート条件の使用

ソート条件は、照会内の行をソートする方法を指定するために使用されます。行は昇順 (A ～ Z) または降順 (Z ～ A) でソートできます。

複数の列で行をソートする場合には、1 番目の列が最初に配列され、 2 番目の列が最初の列の順序の範囲内で配列され、... というようになります。

ソート条件の追加

1. 「指示照会」文書の「ソート条件」セクションで、「**追加**」ボタンをクリックします。

   「ソート条件」ダイアログ・ボックスが開かれます。

2. ソートする列、ソートする方向 (「ソート方向」) を選択して、 「**追加**」をクリックします。ソート条件が指示照会に追加されます。
3. 照会に追加のソート条件を追加して、「**閉じる**」をクリックします。 新規ソート条件をリストした「指示照会」文書が表示されます。

行条件の使用

表内の特定の行だけを何度も表示する場合があります。特定の行を表示するように選択するためには、行条件を追加します。行条件を使用しない場合は、表内のすべての行が表示されます。

使用可能な行条件は次のとおりです。

- Equal to (に等しい)
- Less than (より小さい)
- Less than or equal to (より小さいか、または等しい)
- Greater than (より大きい)
- Greater than or equal to (より大きいか、または等しい)
- Between (の間)
- Starting with (開始)
- Ending with (終了)
- Containing (含む)
- NULL (ヌル)

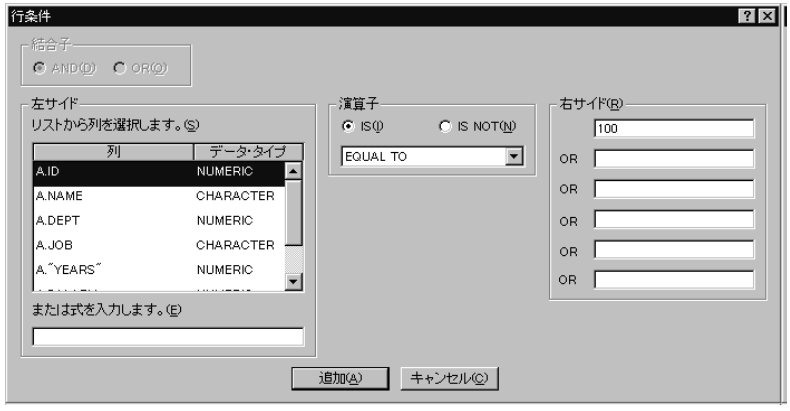
行条件は以下の演算子によって制御されます。

- Is
- Is Not

行条件の追加

1. 「指示照会」文書の「行条件」セクションで、「**追加**」ボタンをクリックします。

「行条件」ダイアログ・ボックスが開かれます。



2. 条件ステートメントの部分を選択して、「**追加**」をクリックします。

行条件の部分	機能
左サイド	検査する列を選択します。
演算子	行の左サイドと右サイドの間の関係を決定します。
右サイド	検査する条件を入力します。

行条件が指示照会に追加されます。

3. 照会に追加の行条件を追加して、「**閉じる**」をクリックします。新規の行条件をリストした「指示照会」文書が表示されます。

指示照会での複数の表の使用

複数の表からの情報を指示照会に組み込むことができます。

1 つまたは複数の結合条件をそれぞれに指定して、 2 つの表を関係付ける必要があります。結合列が等しい表の行だけが、結果に組み込まれます。結合条件内のそれぞれの列のデータ・タイプは、一致していなければなりません。 2 つの列の間の関係が指定されていれば、QMF (Windows 版) はその関係を記憶して、今後の照会でその関係を提示し、それ以降の照会の作成をより簡単に、より効率的にします。

指示照会の結合条件の作成

1. 「指示照会ウィンドウ」の「表」セクションで、少なくとも 2 つの表を追加するために「**追加**」ボタンをクリックします。以前に表が結合されたことがない場合には、「表の結合」ダイアログ・ボックスが開かれます。以前に表が結合されたことがある場合、QMF (Windows 版) は以前に使用された

結合条件を提示します。

2. 各表から同じデータ・タイプの列を選択して、「**追加**」をクリックします。新規の結合条件が指示照会に表示されます。

指示照会と SQL

指示照会インターフェースを使用して、SQL を学習することができます。

指示照会での SQL の表示

指示照会の表示から、ツールバーの「**SQL の表示**」ボタンをクリックします。

指示照会と同等の SQL ステートメントが表示されます。 この表示から、SQL ステートメントを変更することはできません。

指示照会の SQL への変換

指示照会を新規 SQL 照会文書に変換することができます。新規 SQL 照会は、変更、保管、印刷、および実行を行うことができます。「**照会**」メニューから、「**SQL へ変換**」をクリックします。照会は、新規 SQL 照会文書に変換されます。

指示照会の置換変数の使用

置換変数は、SQL 照会内の場合と同じ方法で、指示照会内でも使用できます。『SQL 照会での置換変数』を参照してください。

たとえば、置換変数は次のように使用できます。

- 行条件の場合

```
DEPT Is Greater Than Or Equal To &MinDept
```

- 列の指定の場合

```
&InputNum
```

指示照会の保管

指示照会は、ユーザーの PC、ファイル・サーバー、またはデータベース・サーバーに保管することができます。

ファイルへの指示照会の保管

1. 「指示照会を開く」から、ツールバーの「**保管**」ボタンをクリックします。

注: 照会が以前に保管されている場合も、その照会が再度保管されます。指示照会が以前に保管されていない場合は、「別名保管」ダイアログ・ボックスが開かれます。

2. 指示照会を保管するファイルの名前を入力して、「**OK**」をクリックします。 照会が保管されます。

保管した指示照会ファイルを開く

1. ツールバーの「**開く**」ボタンをクリックします。

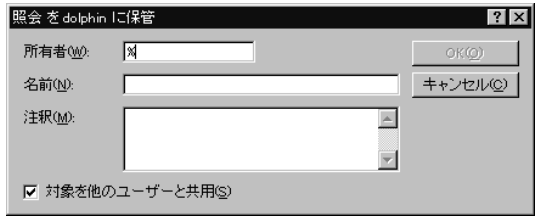
「開く」ダイアログ・ボックスが開かれます。

2. 開きたいファイルを選択して、「**OK**」をクリックします。選択した指示照会が、新規照会文書内に開かれます。

データベース・サーバーへの指示照会の保管

1. 「指示照会を開く」から、ツールバーの「**サーバーに保管**」ボタンをクリックします。

「照会の保管」ダイアログ・ボックスが開かれます。



2. 所有者、名前を入力し、保管されている照会を他のユーザーと共用するのかどうかを選択し、「**OK**」をクリックします。 照会がサーバーに保管されます。

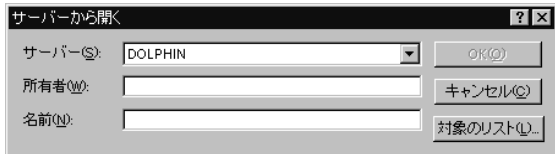
   この名前の照会がすでに存在している場合は、既存の照会を上書きすることを示すプロンプトが出されます。

保管された指示照会をデータベース・サーバーで開く

データベース・サーバーに保管されている指示照会を開くことができます。

1. ツールバーの「**サーバーから開く**」ボタンをクリックします。

「サーバーから開く」ダイアログ・ボックスが開かれます。



2. サーバー、所有者、および名前を入力し、「**OK**」をクリックします。指示照会が開かれます。

指示照会の印刷

指示照会を印刷することができます。また、指示照会の SQL テキストを印刷することもできます。 16ページの『SQL 照会の印刷』を参照してください。

指示照会のプリビュー

指示照会の結果またはテキストを、印刷する前にプリビューする (事前に見る) ことができます。

1. 照会を開いて、指示視点を起動します。照会が表示されます。
2. 「**ファイル**」メニューから、「**ページの設定**」をクリックします。「ページの設定」ダイアログ・ボックスが開かれます。
3. ページのレイアウトに、必要な変更を行い、「**OK**」をクリックします。
4. ツールバーの「**印刷プリビュー**」ボタンをクリックします。

印刷する照会のプリビューが現れます。

第4章 照会結果に関する作業

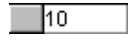
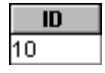
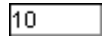
照会結果に対して、書式設定、グループ化、および集約を直接設定することができます。この書式設定は、照会とともに保管したり、書式としてエクスポートすることができます。

照会結果のソートおよびサイズ変更

ユーザーは、照会のデータ結果の選択、サイズ変更、順序変更、およびソートを行うことができます。

列および行の選択

照会が実行されていれば、「結果の表示」にある制御機能を使用して、情報を編集し選択することができます。

列および行のセレクター	外観	機能
行セレクター		クリックして行の中のすべてのデータを選択します。
列セレクター		クリックして列の中のすべてのデータを選択します。
セル		該当のセルを直接クリックしてそれを選択します。
下部および上部へのスクロール・ボタン		クリックして、一組の照会の結果を上部または下部へスクロールします。

列および行のサイズ変更

列および行のサイズを変更して、一組の照会の結果の外観を変更することができます。

1. 「マウス選択」を使用して、 2 つの列の間または 2 つの行の間に黒の区分線を選択します。
2. 区分線を左右に、または上下にドラッグして、列または行のサイズを変更します。

注: 行または列のサイズを変更してから照会の結果を保管すると、その照会は、常に新しい書式設定で表示されます。

列および行の自動サイズ合わせ

列および行のサイズを、そこに入っているデータに自動的に合わせることができます。

マウスを使用して、1 つの列または行の全体を選択し、その列または行と隣接するオブジェクトとの間の黒の区分線をダブルクリックします。これで、その列または行は、データに合うように、自動的にサイズが変更されます。

注: 行または列のサイズを変更してから照会の結果を保管すると、その照会は、常に新しい書式設定で表示されます。

照会の結果のソート

照会が実行されていれば、照会の結果を、アルファベット順に列でソートできます。

照会の「結果の表示」から列を 1 つ選択し、「**結果**」メニューから 「**昇順にソート**」を選択します。

すると、照会の結果は昇順にソートされます。

- あるいは -

照会の「結果の表示」から列を 1 つ選択し、「**結果**」メニューから「**降順にソート**」を選択します。

すると、照会の結果は降順にソートされます。

注: 選択した列に対して、もっと複雑なソートを適用する場合は、「**結果**」メニューから 「**ソート**」を選択してください。

列の順序変更

照会結果の列の順序を変更することができます。

照会の「結果の表示」から、列を 1 つ選択し、それを新しい位置にドラッグします。

すると、その列は新しい順序で表示されます。

照会結果の書式設定

照会および照会の結果を表示するために使用されるフォントを変更することができます。選択できるフォントは、使用しているコンピューターにインストールされているものによって異なります。フォントの追加について詳しくは、使用しているオペレーティング・システムのヘルプ機能を参照してください。

注: 新しい照会結果表示フォントを選択した後で照会を保管すると、その結果は、常に新しいフォントを使用して表示されます。

照会結果表示フォントの選択

1. 「結果の表示」から、「**結果**」メニューの「**フォントの設定**」を選択します。「フォント」ダイアログ・ボックスが開かれます。
2. 照会の結果を表示するためのフォントと活字サイズを選択して、「**OK**」をクリックします。照会結果が、指定された書式で表示されます。

   注: 「**デフォルト設定**」をクリックすると、選択したフォントがすべての照会結果のデフォルト・フォントとして使用されます。

数値照会結果の書式設定

1. 「結果の表示」から、数値を含む列を 1 つ選択し、「**結果**」メニューの「**書式**」を選択します。「書式」ダイアログ・ボックスが開かれます。
2. 適用する書式設定を指定して、「**OK**」をクリックします。値は、行った選択に従って書式設定されます。

   注: 「**デフォルト設定**」をクリックすると、選択したフォントがすべての照会結果のデフォルト・フォントとして使用されます。

書式への照会結果の書式設定の変換

照会結果の書式設定を書式に変換することができます。

1. 「**結果**」メニューから、「**報告書の表示**」を選択します。

   「書式の選択」ダイアログ・ボックスが開かれます。
2. 「**照会から**」を選択して、「**OK**」をクリックします。

   これで、照会結果の書式設定が書式に変換され、「新規書式」ウィンドウの中に開かれます。

照会結果のグループ化および集約

照会結果に対して、グループ化、集約、および要約の書式設定を適用することができます。

照会結果のグループ化

要約情報を付けて、あるいは付けずに、照会結果をグループ化することができます。

1. グループ化する列を選択します。
2. 「**結果**」メニューから、適用したいグループ化のタイプを選択します。

   行った選択に従って、その列がグループ化されます。

照会結果の要約

照会結果を列単位で要約することができます。

1. グループ化する列を選択します。
2. 「**結果**」メニューから、適用したい要約のタイプを選択します。

   行った選択に従って、その列が要約されます。

照会結果の保管および書式設定

照会結果を保管し、書式設定を書式として保管することができます。

照会結果の表としての保管

照会結果を、データベース・サーバーで表として保管することができます。

1. 「**結果**」メニューから、「**データベースに保管**」を選択します。

   「データの保管」ダイアログ・ボックスが開かれます。
2. 所有者および表名を入力して、「**OK**」をクリックします。

   これで、照会結果がデータベースに表として保管されます。

照会結果のファイルへの保管

照会結果を、ユーザーの PC またはファイル・サーバーのファイルに保管することができます。

1. 「**結果**」メニューから、「**ファイルに保管**」を選択します。

   「データのエクスポート」ダイアログ・ボックスが開かれます。
2. ファイルを保管する場所と、必要ならエクスポート・オプションを指定して、「**OK**」をクリックします。

   これで、照会結果がファイルに保管されます。

照会結果の印刷

照会結果のプリビューと印刷を行うことができます。

照会結果のプリビュー

1. 照会を開いて実行します。照会結果が表示されます。
2. 「**ファイル**」メニューから、「**ページの設定**」を選択します。「ページの設定」ダイアログ・ボックスが開かれます。
3. ページのレイアウトに、必要な変更を行い、「**OK**」をクリックします。
4. ツールバーの「**印刷プリビュー**」ボタンをクリックします。

印刷する照会結果のプリビューが表示されます。

照会結果の印刷

1. 照会を開いて「結果表示」をアクティブにします。照会結果が表示されます。
2. 「**ファイル**」メニューから、「**ページの設定**」を選択します。「ページの設定」ダイアログ・ボックスが開かれます。
3. ページのレイアウトに必要な変更を行い、「**OK**」をクリックします。
4. ツールバーの「**印刷**」ボタンをクリックします。

照会結果が印刷されます。

第5章 報告書に関する作業

報告書は、書式からの書式設定と照会結果を結合して作成されます。

書式

書式は、報告書の作成、表示、および印刷に使用される書式設定命令の集まりです。

書式について

書式は、いくつかのコンポーネントから構成されます。書式のコンポーネントは、すべて編集して書式文書にすることができます。

メイン 書式の基本コンポーネント。ヘッダー、フッター、および切れ目が含まれます。

切れ目 報告書の中の 6 行までの小計行の特性、内容、および配置。

計算 報告書の計算式。

注: マシンで書式計算機能を使用するためには、 IBM の Object REXX をインストールしておく必要があります。

列 報告書の列の外観および書式設定。 定義可能な特性には、列の順序、フォーマット、使用法、字下げ、および幅があります。

条件 条件付き書式設定の制約。 たとえば、ある特性と一致しない行を表示しないように書式を設定することができます。

明細 報告書の明細ヘッダーおよび本文テキスト。 ここで、フリー・フォームのテキストを使用して表データを結合または置換することにより、書式文字またはアドレス・ラベルを作成することができます。

最終 報告書の最終テキストの内容および配置。 たとえば、報告書の終わりに最終テキストと要約データを組み込むように選択することができます。

HTML HTML タグと HTML 報告書の書式設定の内容と配置

オプション
報告書のさまざまな外観オプション。

ページ 報告書のページのヘッダーおよびフッターの内容および配置。

書式の使用による報告書の作成

報告書は、書式に書式設定オプションを指定し、照会の結果を結合して作成されます。 このプロセスを繰り返して、単一の照会結果から、複数の報告書を作成することができます。

1. 「照会結果表示」から、ツールバーの「**報告書の表示**」ボタンをクリックします。

「書式の選択」ダイアログ・ボックスが開かれます。

2. 「書式の選択」ダイアログ・ボックスで選択した書式のタイプに応じて、追加の情報を提供するよう求められます。ファイルの場所または所有者と名前、あるいは文書の表題を適宜指定して、「**OK**」をクリックします。選択した書式と現行の照会結果を使用して、報告書が生成されます。

書式の編集

「書式」ウィンドウでは、書式の編集と書式設定のためのオプションが多数用意されています。

「書式を開く」から、「書式」メニューを表示します。「書式」メニューには、書式を編集し、書式設定するためのオプションがすべてリストされています。 これらのコンポーネントは、どれでも、ツールバーの該当のボタンをクリックして編集することができます。

書式の作成

以下に示すステップにはすべて、表 Q.STAFF からのサンプル・データが含まれています。 いろいろな設定を試して、要求に合致したカスタム書式を作成してください。

ステップ 1: 書式を作成する

1. 次の SQL 照会を実行して、報告書に表示するデータを検索します。

   ```
   SELECT * FROM Q.STAFF ORDER BY DEPT, NAME
   ```

   照会結果が表示されます。
2. ツールバーの「**報告書の表示**」ボタンをクリックします。「書式の選択」ダイアログ・ボックスが開かれます。
3. デフォルト書式の使用を指定して、「**OK**」をクリックします。 QMF (Windows 版) は、デフォルト報告書を表示します。デフォルトのフォーマットを変更するためには、ツールバーの書式コンポーネントのボタンのいずれか 1 つをクリックします。各書式コンポーネントのボタンが、書式ツールバー上に表示されます。

ステップ 2: 列の順序を変更する

NAME を報告書の 1 番目の列に、ID を 2 番目の列にするとします。列の順序は、書式の「列」コンポーネントに指定されます。

1. 「**書式**」メニューの「**列...**」をクリックして、「書式」ダイアログ・ボックスの「列」タブを表示します。
2. 既存のシーケンス値を上書きして、列のシーケンスを変更します。 NAME を報告書の 1 番目の列にするには、シーケンス番号 (リストの中で Seq というラベルが付いている列) を 1 に変更します。
3. ID を報告書の 2 番目の列にするには、シーケンス番号を 2 に変更して、「**OK**」をクリックします。 QMF (Windows 版) は、「書式」ウィンドウに新規の列の順序による報告書を表示します。

ステップ 3: 列ヘッダーを変更する

EMPLOYEE を 1 番目の列ヘッダーにし、COMMISSION を最後の列ヘッダーにするとします。列ヘッダー・テキストは、書式の「列」コンポーネントに指定されます。

1. 「**書式**」メニューの「**列...**」をクリックして、「書式」ダイアログ・ボックスの「列」タブを表示します。
2. 既存の列ヘッダー・テキストを上書きして、列ヘッダーを変更します。 1 番目の列ヘッダーを EMPLOYEE に変更し、最後の列ヘッダーを COMMISSION に変更して、「**OK**」をクリックします。 QMF (Windows 版) は、「書式」ウィンドウに新規の列ヘッダーによる報告書を表示します。

ステップ 4: 列の書式を変更する

SALARY 列を、適切な通貨記号を付けて表示するとします。この列のフォーマットは、書式の「列」コンポーネントに指定される編集コードによって決まります。

1. 「**書式**」メニューの「**列...**」をクリックして、「書式」ダイアログ・ボックスの「列」タブを表示します。
2. 既存の編集コードを上書きして SALARY 列の編集コードを D2 に変更し、「**OK**」をクリックします。 QMF (Windows 版) は、SALARY 列に適切な通貨記号が付いた報告書を「書式」ウィンドウに表示します。

ステップ 5: 要約情報を追加する

報告書をセクションに分割して、各部門ごとに別々のセクションを作るとします。さらに、各セクションの最後に各部門の SALARY および COMMISSION の合計を表示するとします。これを行うには、報告書内での各列の使用方法を指定する必要があります。列の使用法は、列の用途コードによって決定され、書式の「列」コンポーネントに指定されます。

1. 「**書式**」メニューの「**列...**」をクリックして、「書式」ダイアログ・ボックスの「列」タブを表示します。
2. DEPT に基づいて報告書をセクションに分割するためには、DEPT の用途コードを BREAK1 に変更します。 BREAK という語で始まる用途コードは、指定した列のセクションの切れ目を作成します。 BREAK という語の後に続く数値によって切れ目レベルが決定されます。1 つの報告書内で最高 6 つまでの切れ目レベルがサポートされます。
3. 各 DEPT に SALARY および COMMISSION の合計の組み込みを指定するために、SALARY および COMMISSION の用途コードを SUM に変更します。
4. 各セクションの切れ目の最後に説明情報も組み込んでおくと、報告書が理解しやすくなります。これを行うには、「**書式**」メニューの「**切れ目...**」をクリックします。
5. 「書式」ダイアログ・ボックスの「切れ目」タブで切れ目のフッター・テキストを指定します。最初の切れ目のフッター行に「部門合計」を設定して、「**OK**」をクリックします。 QMF (Windows 版) は、「書式」ウィンドウを表示します。

ステップ 6: ページのヘッダーとフッターを追加する

報告書にページのヘッダーとフッターを追加するものとします。ページのヘッダーとフッターは、書式の「ページ」コンポーネントに指定されます。

1. 「**書式**」メニューの「**ページ...**」をクリックして、「書式」ダイアログ・ボックスの「ページ」タブを表示します。
2. このダイアログの最上部は、ページ・ヘッダーの指定に使用されます。ページ・ヘッダーの 1 行目を「部門報告書」に設定し、2 行目を「給与および歩合の合計」に設定します。ヘッダーの位置合わせ方法を選択します。
3. このダイアログの最下部は、ページ・フッターの指定に使用されます。ページ・フッターの 1 行目を「ページの終わり」に設定します。 フッターの位置合わせの方法を選択して、「**OK**」をクリックします。 QMF (Windows 版) は、「書式」ウィンドウを表示します。

書式の保管

書式は、ユーザーの PC、ファイル・サーバー、またはデータベース・サーバーに保管することができます。

ファイルへの書式の保管

1. 「書式を開く」から、ツールバーの「**保管**」ボタンをクリックします。
2. 以前に書式が保管されている場合は、「**保管**」を選択します。書式が以前に保管されていない場合は、「別名保管」ダイアログ・ボックスが開かれます。
3. 書式を保管するファイルの名前を入力して、「**OK**」をクリックします。 書式が保管されます。

保管した書式ファイルを開く

1. ツールバーの「**開く**」ボタンをクリックします。

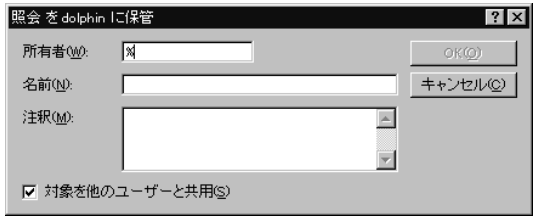
「開く」ダイアログ・ボックスが開かれます。
2. 開くファイルを選択して、「**OK**」をクリックします。選択した書式が新規文書内に開かれます。

データベース・サーバーでの書式の保管

サーバーに保管されている書式は、他のユーザーからアクセスできるようにすることができます。書式を他のユーザーと共用する場合は、書式をデータベース・サーバーに保管してください。

1. 「書式を開く」から、ツールバーの「**サーバーに保管**」ボタンをクリックします。

「書式の保管」ダイアログ・ボックスが開かれます。



2. 所有者と名前を入力し、保管されている書式を他のユーザーと共用するのかどうかを選択し、「**OK**」をクリックします。書式がサーバーに保管されます。

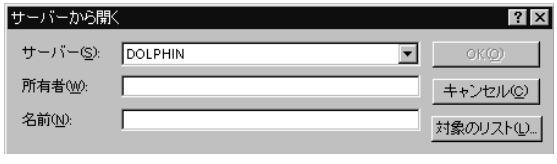
   この名前の書式がすでに存在している場合は、既存の書式を上書きすることを示すプロンプトが出されます。

データベース・サーバーに保管されている書式を開く

ユーザーは、データベース・サーバーに保管されている書式を開くことができます。

1. ツールバーの「**サーバーから開く**」ボタンをクリックします。

「サーバーから開く」ダイアログ・ボックスが開かれます。



2. サーバー、所有者、および名前を入力し、「**OK**」をクリックします。書式が開かれます。

報告書の印刷

ユーザーは、報告書を作成し、印刷することができます。

1. 書式を開き、「**ページの設定**」をクリックします。
2. ページのレイアウトに必要な変更を行い、「**OK**」をクリックします。
3. 「**ファイル**」メニューで「**報告書の印刷**」をクリックします。

   報告書が印刷されます。

報告書のエクスポート

ユーザーは、報告書をファイルへエクスポートすることができます。

1. 書式を開き、「**ページの設定**」をクリックします。
2. ページのレイアウトに必要な変更を行い、「**OK**」をクリックします。
3. 「**ファイル**」メニューで「**報告書のエクスポート**」をクリックします。「報告書のエクスポート」ダイアログ・ボックスが開かれます。

4. 報告書を保管 (保存) するファイルの名前を入力して、「**OK**」をクリックします。報告書がエクスポートされます。

第6章 プロシージャーに関する作業

線形プロシージャーを使用すると、単一のコマンドの実行によって、照会を実行し、報告書を生成し、データを編集し、さらに他の機能を実行することができます。 QMF (Windows 版) がサポートするすべてのプロシージャー・コマンドの完全なリストについては、オンライン・ヘルプ機能を参照してください。

ロジックを持つプロシージャー (つまり、REXX プロシージャー) は、線形プロシージャーに似ていますが、これには、プロシージャー・コマンドだけでなく IBM の Object REXX プログラム言語が含まれています。ロジックを持つプロシージャーを実行するためには、Object REXX をローカルにインストールしておくことが必要です。

プロシージャーの実行

プロシージャーは、1 つのコマンドを使用して、複数の機能を実行するために使用されます。

新規の線形プロシージャーの作成

「**ファイル**」メニューから、「**新規プロシージャー**」を選択します。

新規のプロシージャー文書が開かれます。

新規のロジックを持つプロシージャーの作成

1. 「**ファイル**」メニューから、「**新規プロシージャー**」を選択します。
   新規のプロシージャー文書が開かれます。
2. プロシージャーの先頭行として、REXX コメント行を入力します。 REXX コメント行は、/* で始まり、*/ で終わります。
3. 希望する QMF プロシージャー・コマンドがあれば、プロシージャー内に入力します。 QMF コマンドは、大文字で入力し、引用符で囲むことが必要です。
4. 希望する任意の REXX コマンドがあれば、プロシージャー内に入力します。

注: REXX コマンドは、ローカルで実行され、データベース・サーバーでは実行されません。オブジェクト REXX をローカルにインストールしておくことが必要です。

データベース・サーバーでのプロシージャーの実行

1. 新規のプロシージャー文書を開いて一組のコマンドを入力するか、あるいはファイルまたはデータベース・サーバーから既存のプロシージャーを開きます。
2. ツールバーの「**プロシージャー実行**」ボタンをクリックします。

該当のプロシージャーが実行されます。

プロシージャーの保管

プロシージャーは、ユーザーの PC、ファイル・サーバー、またはデータベース・サーバーに保管することができます。

ファイルへのプロシージャーの保管

1. 「プロシージャーを開く」から、ツールバーの「**保管**」ボタンをクリックします。

プロシージャーが以前に保管されていても、プロシージャーは保管されます。プロシージャーが以前に保管されていない場合は、「別名保管」ダイアログ・ボックスが開かれます。
2. プロシージャーを保管するファイルの名前を入力して、「**OK**」をクリックします。 プロシージャーが保管されます。

保管されたプロシージャー・ファイルを開く

1. ツールバーの「**開く**」ボタンをクリックします。

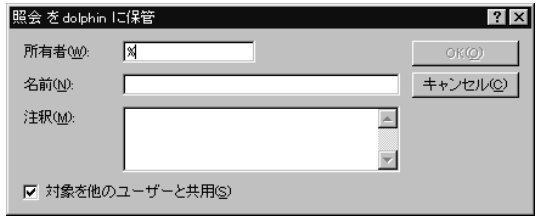
「開く」ダイアログ・ボックスが表示されます。

2. 開くファイルを選択して、「**OK**」をクリックします。選択したプロシージャーが、新規プロシージャー文書内で開かれます。

データベース・サーバーへのプロシージャーの保管

1. 「プロシージャーを開く」から、ツールバーの「**サーバーに保管**」ボタンをクリックします。

「プロシージャーの保管」ダイアログ・ボックスが開かれます。



2. 所有者と名前を入力し、保管されているプロシージャーを他のユーザーと共用するのかどうかを選択し、「**OK**」をクリックします。プロシージャーは、サーバーに保管されます。

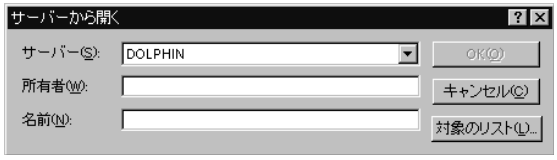
   この名前のプロシージャーがすでに存在している場合は、前から存在しているプロシージャーを上書きすることを示すプロンプトが出されます。

データベース・サーバーに保管されているプロシージャーを開く

データベース・サーバーに保管されているプロシージャーを開くことができます。

1. ツールバーの「**サーバーから開く**」ボタンをクリックします。

「サーバーから開く」ダイアログ・ボックスが開かれます。



2. サーバー、所有者、および名前を入力し、「**OK**」をクリックします。該当のプロシージャーが開かれます。

プロシージャーの印刷

プロシージャーのテキストを印刷することができます。

プロシージャーのプリビュー

1. プロシージャーを開きます。プロシージャー・コマンドが現れます。
2. 「**ファイル**」メニューから、「**ページの設定**」をクリックします。「ページの設定」ダイアログ・ボックスが開かれます。
3. ページのレイアウトに、必要な変更を行い、「**OK**」をクリックします。
4. ツールバーの「**印刷プリビュー**」ボタンをクリックします。

印刷するプロシージャーのプリビューが現れます。

プロシージャーの印刷

1. プロシージャーを開きます。プロシージャー・コマンドが現れます。
2. 「**ファイル**」メニューから、「**ページの設定**」をクリックします。「ページの設定」ダイアログ・ボックスが開かれます。
3. ページのレイアウトに、必要な変更を行い、「**OK**」をクリックします。
4. ツールバーの「**印刷**」ボタンをクリックします。

プロシージャーが印刷されます。

第7章 リストに関する作業

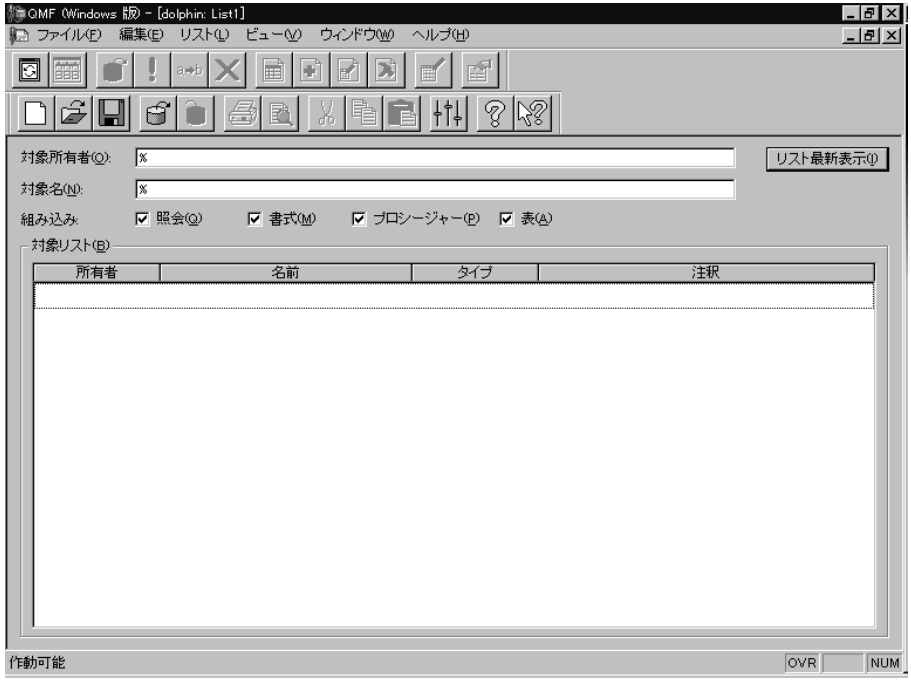
「リスト」は、QMF オブジェクトの集合を表示するための簡単な方法を提供します。

オブジェクト

QMF (Windows 版) は、4 つのタイプのオブジェクト、すなわち、照会、書式、プロシージャー、および表を認識します。「リスト」ウィンドウを使用して、オブジェクト名、所有者、およびタイプに基づいて、オブジェクトを表示することができます。

オブジェクトのリスト

1. 「**ファイル**」メニューから、「**新規リスト**」を選択します。
   「リスト」ウィンドウが開かれます。



2. 所有者と名前を指定します。

注: マッチングするオブジェクトのリストからオブジェクトを選択する際に、パターンを使用することができます。

- 任意の文字を含む任意の長さのストリングのマッチングを行うためには、パーセント文字 (%) を使用します。 たとえば、名前が文字 A で始まる表をすべてリストするためには A% と入力します。
- 単一文字のマッチングを行うためには、下線文字 (_) を使用します。たとえば、2 番目の位置に文字 A が入っている所有者の表をすべてリストするためには _A% と入力します。

3. 検索しているオブジェクトのタイプを選択します。
4. 「**リスト最新表示**」をクリックします。データベース・サーバーに保管されていて、条件にマッチングしているオブジェクトのリストが表示されます。

リスト・ウィンドウのコマンド

「リスト」ウィンドウ内のオブジェクトを右クリックすると、「リスト」メニューのコマンドと同じコマンドのリストが表示されます。

オブジェクトの表示
選択したオブジェクトを開いて表示します。照会、書式、プロシージャー、および表に使用することができます。

オブジェクトの実行
選択したオブジェクトを実行します。照会およびプロシージャーに使用することができます。

オブジェクトのドロー
選択した表に基づいて照会を作成します。 SQL SELECT 照会、SQL UPDATE 照会、SQL INSERT 照会、または指示照会をドローするように選択することができます。表に使用することができます。

オブジェクトの編集
選択したオブジェクトを開いて編集します。表に使用することができます。

プロパティー
コメント、属性および使用に関するヒストリー情報も含めて、選択されたオブジェクトのプロパティーを表示します。照会、書式、プロシージャー、および表に使用することができます。

リストの作成

オブジェクトの集合としての役割を果たすリストを作成することができます。たとえば、在庫に関係するすべての照会、書式、プロシージャー、および表のリストを作成して、作業を 1 つの場所にまとめておくことができます。このリストを作成したら、このリストへのオブジェクトの追加や除去を行ったり、さらに将来の使用に備えてリストを保管することができます。

リストへのオブジェクトの追加

リストにオブジェクトを追加することができます。

開いたリストから、追加するオブジェクトの所有者と名前の情報を指定して、ツールバーの「**リストに追加**」ボタンをクリックします。

所有者と名前の条件にマッチングしたオブジェクトがリストに追加されます。

リストからのオブジェクトの除去

リストから関係のないオブジェクトを除去することができます。

「リストを開く」から、ツールバーの「**除去**」ボタンをクリックします。

該当のオブジェクトは、リストから除去されますが、削除はされません。

ファイルへのリストの保管

1. 「リストを開く」から、ツールバーの「**保管**」ボタンをクリックします。

   リストが以前に保管されていても、リストは保管されます。リストが以前に保管されていない場合は、「別名保管」ダイアログ・ボックスが開かれます。
2. リストを保管するファイルの名前を入力して、「**OK**」をクリックします。リストが保管されます。

保管されたリスト・ファイルを開く

1. ツールバーの「**開く**」ボタンをクリックします。

   「開く」ダイアログ・ボックスが開かれます。
2. 開くファイルを選択して、「**OK**」をクリックします。選択されたリストが、リスト文書の中で開かれます。

第8章 ジョブ・ファイルに関する作業

ジョブ・ファイルを使用して、プロシージャーをスケジュールし、実行することができます。ジョブ・ファイルは、Windows スケジューラーを使用して、事前設定された時刻および日付にしたがって、プロシージャーを実行します。

ジョブ・ファイル

ジョブ・ファイルを作成し、それをローカルで保管するか、またはデータベース・サーバーに保管することができます。

ジョブ・ファイルの作成

1. 「**ファイル**」メニューから、「**新規ジョブ**」を選択します。
   新規の「ジョブ文書」が開かれます。

ジョブ・ファイルの実行

ローカルに保管されているジョブ・ファイルを実行することができます。

1. ジョブ・ファイルを開きます。
2. ツールバーの「**ジョブ実行**」ボタンをクリックします。

3. 区分線を左右に、または上下にドラッグして、列または行のサイズを変更します。

   注: 行または列のサイズを変更してから照会の結果を保管すると、その照会は、常に新しい書式設定で表示されます。

列および行の自動サイズ合わせ

列および行のサイズを、そこに入っているデータに自動的に合わせることができます。

マウスを使用して、1 つの列または行の全体を選択し、その列または行と隣接するオブジェクトとの間の黒の区分線をダブルクリックします。これで、その列または行は、データに合うように、自動的にサイズが変更されます。

注: 行または列のサイズを変更してから照会の結果を保管すると、その照会は、常に新しい書式設定で表示されます。

照会の結果のソート

照会が実行されていれば、照会の結果を、アルファベット順に列でソートできます。

照会の「結果の表示」から列を 1 つ選択し、「**結果**」メニューから 「**昇順にソート**」を選択します。

すると、照会の結果は昇順にソートされます。

- あるいは -

照会の「結果の表示」から列を 1 つ選択し、「**結果**」メニューから「**降順にソート**」を選択します。

すると、照会の結果は降順にソートされます。

注: 選択した列に対して、もっと複雑なソートを適用する場合は、「**結果**」メニューから 「**ソート**」を選択してください。

列の順序変更

照会結果の列の順序を変更することができます。

照会の「結果の表示」から、列を 1 つ選択し、それを新しい位置にドラッグします。

すると、その列は新しい順序で表示されます。

照会結果の書式設定

照会および照会の結果を表示するために使用されるフォントを変更することができます。選択できるフォントは、使用しているコンピューターにインストールされているものによって異なります。フォントの追加について詳しくは、使用しているオペレーティング・システムのヘルプ機能を参照してください。

注: 新しい照会結果表示フォントを選択した後で照会を保管すると、その結果は、常に新しいフォントを使用して表示されます。

照会結果表示フォントの選択

1. 「結果の表示」から、「**結果**」メニューの「**フォントの設定**」を選択します。「フォント」ダイアログ・ボックスが開かれます。
2. 照会の結果を表示するためのフォントと活字サイズを選択して、「**OK**」をクリックします。照会結果が、指定された書式で表示されます。

   注: 「**デフォルト設定**」をクリックすると、選択したフォントがすべての照会結果のデフォルト・フォントとして使用されます。

数値照会結果の書式設定

1. 「結果の表示」から、数値を含む列を 1 つ選択し、「**結果**」メニューの「**書式**」を選択します。「書式」ダイアログ・ボックスが開かれます。
2. 適用する書式設定を指定して、「**OK**」をクリックします。値は、行った選択にしたがって、書式設定されます。

   注: 「**デフォルト設定**」をクリックすると、選択したフォントがすべての照会結果のデフォルト・フォントとして使用されます。

書式への照会結果の書式設定の変換

照会結果の書式設定を書式に変換することができます。

1. 「**結果**」メニューから、「**報告書の表示**」を選択します。

   「書式の選択」ダイアログ・ボックスが開かれます。
2. 「照会から」を選択して、「**OK**」をクリックします。

   これで、照会結果の書式設定が書式に変換され、「新規書式」ウィンドウの中に開かれます。

照会結果のグループ化および集約

照会結果に対して、グループ化、集約、および要約の書式設定を適用することができます。

照会結果のグループ化

要約情報を付けて、あるいは付けずに、照会結果をグループ化することができます。

1. グループ化する列を選択します。
2. 「**結果**」メニューから、適用したいグループ化のタイプを選択します。

   行った選択に従って、その列がグループ化されます。

照会結果の要約

照会結果を列単位で要約することができます。

1. グループ化する列を選択します。
2. 「**結果**」メニューから、適用したい要約のタイプを選択します。

   行った選択に従って、その列が要約されます。

照会結果の保管および書式設定

照会結果を保管し、書式設定を書式として保管することができます。

照会結果の表としての保管

照会結果を、データベース・サーバーで表として保管することができます。

1. 「**結果**」メニューから、「**データベースに保管**」を選択します。

   「データの保管」ダイアログ・ボックスが開かれます。
2. 所有者および表名を入力して、「**OK**」をクリックします。

   これで、照会結果がデータベースに表として保管されます。

照会結果のファイルへの保管

照会結果を、ユーザーの PC またはファイル・サーバーのファイルに保管することができます。

1. 「**結果**」メニューから、「**ファイルに保管**」を選択します。

   「データのエクスポート」ダイアログ・ボックスが開かれます。
2. ファイルを保管する場所と、必要ならエクスポート・オプションを指定して、「**OK**」をクリックします。

   これで、照会結果がファイルに保管されます。

照会結果の印刷

照会結果のプリビューと印刷を行うことができます。

照会結果のプリビュー

1. 照会を開いて実行します。照会結果が表示されます。
2. 「**ファイル**」メニューから、「**ページの設定**」を選択します。「ページの設定」ダイアログ・ボックスが開かれます。
3. ページのレイアウトに、必要な変更を行い、「**OK**」をクリックします。

4. ツールバーの「**印刷プリビュー**」ボタンをクリックします。

印刷する照会結果のプリビューが表示されます。

照会結果の印刷

1. 照会を開いて「結果表示」をアクティブにします。照会結果が表示されます。
2. 「**ファイル**」メニューから、「**ページの設定**」を選択します。「ページの設定」ダイアログ・ボックスが表示されます。
3. ページのレイアウトに任意の必要な変更を行い、「**OK**」をクリックします。
4. ツールバーの「**印刷**」ボタンをクリックします。

照会結果が印刷されます。

第9章 静的照会に関する作業

静的照会とは、前にデータベース・サーバーに渡され、パッケージの形にバインドされている SQL 照会のことです。静的照会が実行されるとき、データベース・サーバーは、現在、照会ウィンドウに表示されている SQL テキストではなく、パッケージの形にバインドされている SQL テキストを使用します。静的照会は、動的照会よりもリソースの使用の点で効率がよいが、静的照会は編集することはできません。

静的照会

静的照会は、以前から存在している SQL および指示照会から作成されます。

静的照会の作成

1. 「**照会**」メニューから、「**静的パッケージのバインド**」を選択します。「静的パッケージのバインド」ダイアログ・ボックスが開かれます。

2. 「パッケージ」タブを選択して、集合 ID およびパッケージ名を入力し、その他の必要なオプションを変更します。
3. 照会に置換変数が含まれている場合は、「変数の入力」タブを選択します。置換変数をホスト変数で置き換えます。

4. 「**OK**」をクリックします。静的照会がバインドされます。

   注: 照会をバインドした後、その照会をファイルまたはデータベース・サーバーのいずれかに保管する必要もあります。

置換変数のホスト変数への置換

パッケージをバインドする場合は、SQL テキスト内の各置換変数の代わりに使用するホスト変数を指定する必要があります。ただし、置換変数は、常にホスト変数によって直接に置換できるとは限りません。テキストがデータベース・サーバーに送信される前に、置換変数によって照会テキスト内で直接テキスト置換が行われます。ホスト変数は、データベース・サーバーへの照会の部分として送信されます。ホスト変数を照会内で使用できるようにする方法および場所の規則については、使用しているデータベース・サーバーの資料を参照してください。

置換変数とホスト変数の間の関係がいったん指定されると、QMF (Windows 版) はその関係を覚えておき、今後の照会でその関係を提示し、パッケージのバインディングをより簡単に行えるようにします。

ホスト変数の有効なデータ・タイプは次のとおりです。

- CHAR(n)
- VARCHAR(n)
- INTEGER
- SMALLINT
- FLOAT
- DECIMAL(p,s)
- DATE
- TIME
- TIMESTAMP

1. 「静的パッケージのバインド」ダイアログ・ボックスから、「変数の入力」タブを選択します。

2. 各ホスト変数の変数タイプを入力して、「**OK**」をクリックします。 置換変数がホスト変数に変換されます。

静的照会の実行

静的照会は、他の照会と同様に実行できます。 9ページの『SQL 照会』を参照してください。

第10章 表編集プログラムに関する作業

SQL ステートメントを書くのではなしに、表編集プログラムを使用して、表に保管されているデータを検索し、追加し、編集し、削除することができます。

表編集プログラム

表編集プログラムを使用すると、データの編集および検索に柔軟性が得られます。

表編集プログラムの使用による行の検索

1. 「**ファイル**」メニューから、「**表編集プログラム**」を選択します。「表編集」ダイアログ・ボックスが開かれます。

2. 表を指定します。

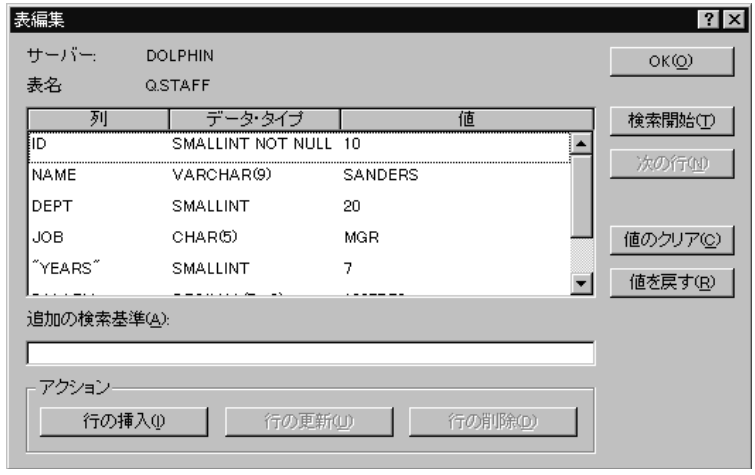
   注: マッチングする表のリストから表名を選択する際に、パターンを使用することができます。

   - 任意の文字を含む任意の長さのストリングのマッチングを行うためには、パーセント文字 (%) を使用します。 たとえば、名前が文字 A で始まる表をすべてリストするためには A% と入力します。
   - 単一文字のマッチングを行うためには、下線文字 (_) を使用します。たとえば、2 番目の位置に文字 A が入っている所有者の表をすべてリストするためには _A% と入力します。

   パターンを入力した後、「**表のリスト**」をクリックして、結果リストから表を選択します。

3. 「保管モード」を選択します。
   - 「即時」を選択すると、表は、変更されるたびに即時にデータベース・サーバーで更新されます。

- 「終了時」を選択すると、表は、すべての変更の入力完了後にデータベース・サーバーで更新されます。表の変更を行っている途中では、他のユーザーはその表を変更することはできません。



4. 「**編集**」をクリックします。「表編集」ダイアログ・ボックスが開かれます。
5. 「値」列に検索する値を入力するか、または「追加の検索基準」フィールドに検索基準を入力して、さらに複雑な検索条件を指定します。 「追加の検索基準」フィールドには、任意の有効な SQL 述部を入力することができます。
6. 「**検索開始**」をクリックします。最初に一致した行が「値」列に表示されます。

行の追加

1. 「表編集」ダイアログ・ボックスに、新規レコードに関する情報を入力します。
2. 「**行の挿入**」をクリックします。新規の行が表に追加されます。
3. 「**OK**」をクリックします。変更が保管されます。

行の変更

1. 「表編集」ダイアログ・ボックスから、変更する行を検索します。
2. 変更する行が表示されるまで、「**次の行**」をクリックします。
3. 「値」列内のデータを編集して、「**行の更新**」をクリックします。 行が更新されます。

4. 「**OK**」をクリックします。変更が保管されます。

行の削除

1. 「表編集」ダイアログ・ボックスから、削除する行を検索します。
2. 削除する行が表示されるまで、「**次の行**」をクリックします。
3. 「**行の削除**」をクリックします。行が削除されます。
4. 「**OK**」をクリックします。変更が保管されます。

照会結果の表示からの表の編集

照会結果の表示から、直接に表を編集することができます。

照会結果の表示からの行の削除

照会結果の表示にある表から、個々の行を削除することができます。

照会結果の表示から行を選択し、「**編集**」メニューの「**削除**」を選択します。行が削除されます。

照会結果の表示からの列の更新

照会結果の表示の中の個々の列の内容を更新することができます。

照会結果の表示から、セルをダブルクリックし、新規の値を入力し、Enter キーを押します。 表は更新されます。

DB2 書式

マシンに DB2 Forms の User コンポーネントをインストールしてある場合には、それを、LOB データが含まれていない表の表編集プログラムとして使用することができます。 DB2 Forms の詳細については、www.rocketsoftware.com/db2forms にある DB2 Forms の Resource Center を参照してください。

第11章 データの分散

データを他のデータベースおよびアプリケーションにエクスポートすることができます。

データのエクスポート

次のようにして、データを QMF (Windows 版) から他のアプリケーションへエクスポートすることができます。

- データをテキスト、CSV、IXF、または HTML のファイルへエクスポートする。
- 照会結果を表に保管する。
- 照会結果を、直接、Microsoft Excel のスプレッドシートに追加する。

ファイルへのデータのエクスポート

1. 照会結果を表示しているときに、「**ファイル**」メニューの「**データのエクスポート**」を選択します。「データのエクスポート」ダイアログ・ボックスが開かれます。

2. 該当の「出力ファイルの種類」を選択して、「**オプション**」ボタンをクリックします。選択する出力ファイルの種類に応じて、「テキスト / DEL エクスポート・オプション」ダイアログ・ボックス、「HTML エクスポート・

オプション」ダイアログ・ボックス、「IXF エクスポート・オプション」ダイアログ・ボックス、あるいは、「CSV エクスポート・オプション」ダイアログ・ボックスが表示されます。

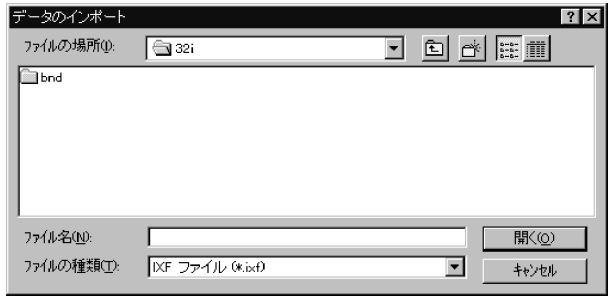
- .TXT という拡張子の付いたテキスト・ファイルを作成することができます。これは、(「テキスト / DEL エクスポート・オプション」ダイアログ・ボックスに指定されたとおりの) オプションのストリングおよび列区切り文字をもった標準の ASCII ファイルです。
- .HTM という拡張子を持つ HTML ファイルを作成することができます。これは、どの Web ブラウザーでも表示することができる HTML ファイルです。 HTML タグはすべて、ファイルの中で自動的に生成され、インターネットまたはイントラネットの Web サイトで発行できるようになっています。「HTML のエクスポート・オプション」ダイアログ・ボックスで選択したオプションによって、エクスポートしたデータの外観が制御されます。
- .IXF ファイルを作成することができます。 IXF エクスポートでは、列ヘッダーおよびデータ・タイプも含めて、データベース情報がすべて保存されます。 このエクスポートは、通常、あるデータベースから別のデータベースへ情報を転送するために使用されます。
- .CSV ファイルを作成することができます。 CSV エクスポートは、コンマを列区切り文字として使用しており、テキスト・エクスポートと非常によく似ています。 このフォーマットは、スプレッドシート・アプリケーションで普通に用いられます。

3. エクスポート・ファイルの選択したタイプに対するオプションを選択して、「**OK**」をクリックします。 「オプション」ダイアログ・ボックスが閉じます。
4. 「データのエクスポート」ダイアログ・ボックスの「**OK**」をクリックします。データがエクスポートされます。

データのインポート

IXF ファイルに保管されているデータをインポートすることができます。データは、照会ウィンドウにインポートされると、データベース・サーバーに保管され、新しいファイルへエクスポートされるか、あるいは報告書用に使用されます。 PC/IXF および文字モードのシステム /370 IXF ファイルがサポートされています。

1. 「**ファイル**」メニューから、「**データのインポート**」を選択します。「データのインポート」ダイアログ・ボックスが開きます。

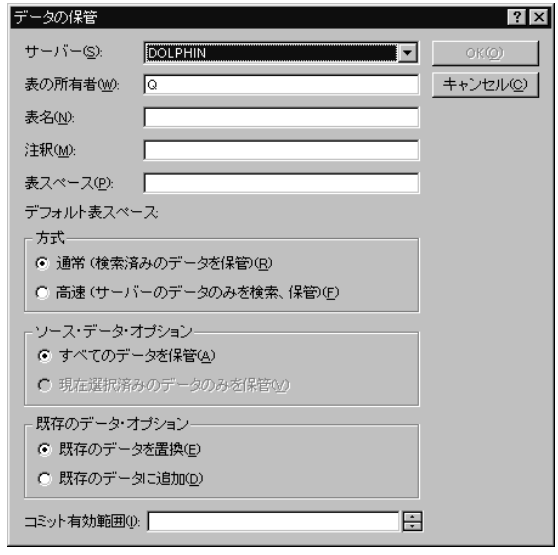


2. インポートするファイルを選択して、「**OK**」をクリックします。インポートされたデータが、新しいの照会ウィンドウに表示されます。

データベース・サーバーへのデータの保管

インポートされた照会結果は、データベース表に保管することができます。

1. インポートした照会結果を表示しているときに、「**ファイル**」メニューの「**データの保管**」を選択します。「データの保管」ダイアログ・ボックスが開かれます。



2. データベース・サーバーを選択し、表の所有者および名前を入力し、その他の必要なオプションを選択して、「**OK**」をクリックします。データが保管されます。

Send To コマンドの使用

QMF (Windows 版) には、Send To コマンドと基本的な電子メール・クライアントが組み込まれています。ジョブ・ファイルと一緒に Send To コマンドを使用して、照会をスケジュールし、その結果を配布することができます。

1. 「**ファイル**」メニューから、「**送信**」および 「**インターネット・メール宛先**」を選択します。「メッセージ」ダイアログ・ボックスが開かれます。
2. メッセージの宛先、件名、メッセージのテキストを指定して、「**次へ**」をクリックします。「接続」ダイアログ・ボックスが開かれます。
3. メッセージに対する接続の追加または除去を行って、「**次へ**」をクリックします。「メッセージ送信」ダイアログ・ボックスが開かれます。
4. 自分のメール・サーバーの名前を指定して、「**終了**」をクリックします。これで、メッセージが送信されます。

Microsoft Excel アドインの使用

QMF (Windows 版) には、Microsoft Excel 7.0 またはそれ以降用のアドインが組み込まれています。これらのアドインによって、Excel から QMF (Windows 版) を実行して、照会結果を直接スプレッドシートに戻すことができます。これらのアドインは、「標準インストール (Typical installation)」オプションを選択するか、または「カスタム・インストール (Custom installation)」オプションを選択して「Microsoft Excel アドイン (Microsoft Add-In)」オプションを選択すると、自動的にインストールされます。

1. Excel ツールバーの「**QMF (Windows 版)**」ボタンをクリックします。

   QMF (Windows 版) が開かれます。
2. QMF (Windows 版) から、照会を選択して実行します。照会結果が表示されます。
3. Excel に戻すデータを選択します。
4. 「**ファイル**」メニューから、**「Microsoft Excel へデータを戻す」**を選択します。 Excel が開かれ、「QMF (Windows 版) アドイン」ダイアログ・ボックスが表示されます。
5. データの宛先範囲を入力して、「**OK**」をクリックします。データがスプレッドシートに追加されます。

サンプル・アプリケーションの使用

QMF (Windows 版) では、いくつかのサンプル・アプリケーションおよびアドイン「組み込みソリューション」を使用することができます。詳細については、http://www.ibm.com/qmf/ の IBM Web サイトを参照してください (英語版のみ)。

第12章 QMF レポート・センターの使用

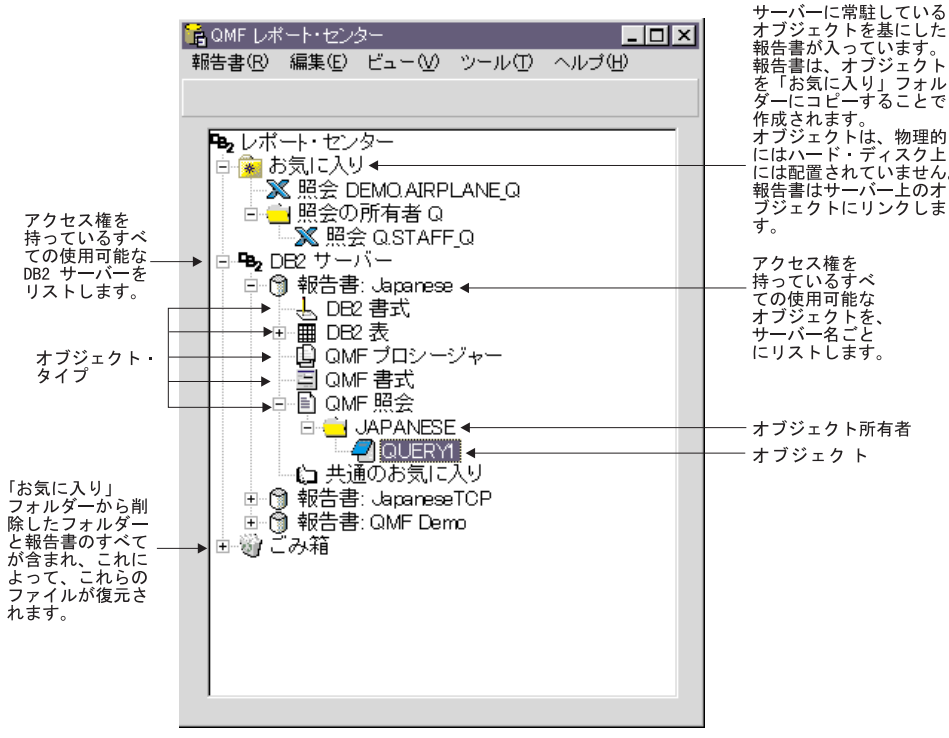
QMF レポート・センターにより、共用の QMF 照会、書式、プロシージャー、および表を使用して、カスタム報告書を作成することができます。これらのオブジェクトに高速アクセスすることで、好みのデータ・フォーマットを指定し、さまざまなアプリケーションでの表示および取り扱いが可能なカスタム報告書を作成することができます。

QMF レポート・センターの開始

- いずれかのオブジェクトまたはフォルダーで右マウス・ボタン・クリックして、ツールバーのメニューから使用可能なものと同じオプションをアクティブにします。
- いずれかのフォルダーの隣にあるプラス記号 **(+)** をクリックして、第 1 レベルの内容を開きます。 SHIFT キーを押したままプラス記号 **(+)** をクリックして、そのフォルダーの下にあるすべてのレベルを開きます。

「QMF レポート・センター」ウィンドウ

「QMF レポート・センター」ウィンドウには、ツリーに似た構造で、使用可能なお気に入り、DB2 サーバー、共通のお気に入り、オブジェクト、およびごみ箱が含まれています。

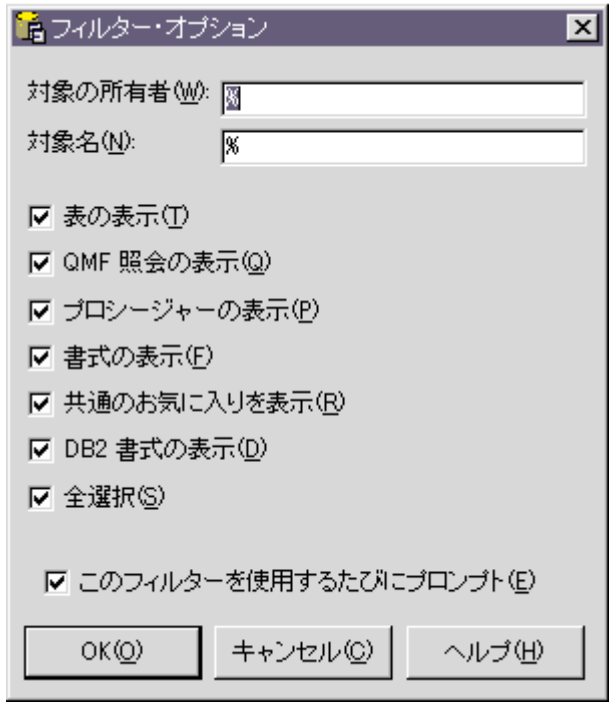


このウィンドウに表示されているオブジェクトには、そのオブジェクトの出力が関連付けられているアプリケーションのタイプを表すアイコンが含まれています。

サーバーへの接続

1. DB2 サーバーの下にサーバー名が何も表示されていない場合には、プラス記号 **(+)** をクリックします。

2. サーバーの隣にあるプラス記号 **(+)** をクリックします。「フィルター・オプション」ダイアログ・ボックスが開かれます。



3. 表示したいオブジェクト・タイプを選択して、「**OK**」をクリックします。サーバー上で使用可能なオブジェクトが表示され、オブジェクト・タイプによってグループ化されます。

報告書およびオブジェクトに関する作業

報告書は、QMF オブジェクトを基にしています。ユーザーの個人的な「お気に入り」フォルダーおよび「共通のお気に入り」フォルダーに入っている項目はすべて報告書と見なされます。書式設定を操作して、これらの項目についてのオプションを表示することができます。これらの「お気に入り」フォルダーに入っている項目は、サーバーに常駐している QMF オブジェクトにリンクします。 実際には QMF オブジェクトを変更するのではなく、報告書として参照されているオブジェクトへのリンクを変更します。報告書はオブジェクトを基にしているため、オブジェクトの特性が報告書にも適用されます。

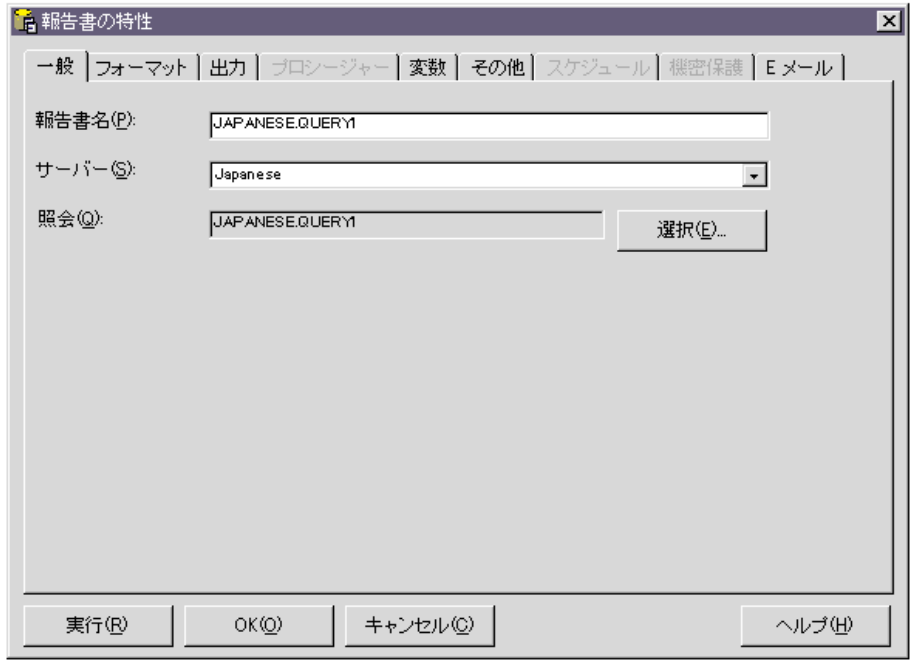
サーバーに常駐しているオブジェクトから報告書を作成することはできますが、報告書はサーバーには保管されません。この機能により、一回限りの報告

書を迅速に作成することができます。ただし、サーバー上のオブジェクトから報告書を作成した後、その報告書を「お気に入り」フォルダーに保管するオプションが提供されています。

報告書の実行

「お気に入り」フォルダーから、もしくはサーバー上にあるオブジェクトから、報告書を実行することができます。

1. 選択した報告書またはオブジェクトで、「報告書」メニューの「*特性*」を選択します。「報告書の特性」ダイアログ・ボックスが開かれます。



2. 必要に応じて、特性を定義します。
3. 「**実行**」ボタンをクリックします。報告書が処理されます。さらに、「報告書の特性の出力」ダイアログ・ボックスで「*発行後に報告書を表示*」オプションを選択した場合には、指定されたアプリケーションで報告書が表示されます。

また、以下のいずれかの方法を用いると、報告書を迅速に実行することができます。

- 報告書を選択して、「報告書」メニューから「*実行*」を選択します。
- 報告書で右マウス・ボタンをクリックしてから、「*実行*」を選択します。

- 報告書名をダブルクリックします。

フォルダーおよびお気に入りに関する作業

フォルダーは、報告書および QMF オブジェクトをグループ化する際に使用されます。フォルダーは、オブジェクト所有者の名前にしたがって名前が付けられます。報告書の実行や報告書の特性の定義など、報告書に対して実行する操作と同じ操作をフォルダーに対して実行することができます。フォルダーに対するこれらの操作の実行は、そのフォルダー内に含まれているどの報告書に対しても適用されます。たとえば、フォルダー内に含まれているすべての報告書を連続して実行したい場合には、そのフォルダーを選択してから、「報告書」メニューの「**実行**」を選択します。

QMF レポート・センターには、報告書を保管することができる最上位のフォルダーが 2 つあります。このフォルダーには、サーバー上のオブジェクトを指す報告書が入っています。オブジェクト自体は、「お気に入り」フォルダー内には入っていません。個人的な「お気に入り」フォルダーは、ローカルに (そのユーザーの PC 上に) 常駐するため、このフォルダーとその内容には、他のユーザーはアクセスできません。「共通のお気に入り」フォルダーは、サーバー上に常駐するため、すべての許可ユーザーがアクセスできます。リソース限界に応じて、複数の「共通のお気に入り」フォルダーにアクセスすることは可能ですが、各サーバー上に複数の「共通のお気に入り」フォルダーが存在することはありません。

QMF オブジェクトを「お気に入り」フォルダーにコピーすると、そのフォルダーは自動的に名前が変更されて、オブジェクト・タイプと所有者名が組み込まれます。オブジェクト・タイプ全体 (すなわち、同一タイプのオブジェクトのフォルダー) をサーバー (たとえば、全照会) からコピーすると、そのサーバー名は新規のフォルダー名にも組み込まれます。

お気に入りへの報告書の追加

サーバーからオブジェクトまたは報告書を、個人的な「お気に入り」フォルダーあるいはサーバー上の「共通のお気に入り」フォルダーに追加することができます (システム管理者から許可を与えられている場合)。

報告書を個人的なお気に入りに追加する場合:

選択した報告書またはオブジェクトで、「報告書」メニューの「お気に入りに*追加*」を選択するか、もしくは、その報告書またはオブジェクトを個人的な「お気に入り」フォルダーにドラッグします。命名規則

ObjecttypeOWNERNAME.OBJECTNAME に従って、個人的な「お気に入り」フォルダーの先頭に報告書が追加されます。

報告書を共通のお気に入りに追加する場合:

QMF オブジェクトまたは報告書を、サーバー上の「共通のお気に入り」フォルダーにドラッグします。個人的な「お気に入り」フォルダーもしくは任意のサーバーから、報告書を追加することができます。

注: 共通のお気に入りに追加する場合、あるいは共通のお気に入り内の報告書を変更する場合は、 更新内容がサーバーに保管される前に、「報告書」メニューから 「*変更を共通のお気に入りに保管*」を選択することが必要です。

QMF レポート・センターの使用法に関する詳細は、オンライン・ヘルプ・システムを参照してください。

第13章 QMF (Windows 版) API の使用

QMF (Windows 版) アプリケーション・プログラミング・インターフェースを使用して、カスタム・アプリケーションを作成することができます。

API を介した QMF (Windows 版) の制御

以下のステップは、QMF (Windows 版) を制御するために API に関してどのように作業するかについての概要を示しています。

1. QMF (Windows 版) API オブジェクトのインスタンスを作成します。Microsoft Visual Basic を使用している場合には、QMF (Windows 版) タイプのライブラリー qmfwin.tlb にリファレンスを追加します。その後、次のように Dim ステートメントを使用します。

   ```
   Dim QMFWin As New QMFWin
   ```

   あるいは、次のように CreateObject ステートメントを使用します。

   ```
   Dim QMFWin As Object
   Set QMFWin = CreateObject ("QMFWin.Interface")
   ```

   注: 異なる開発環境を使用している場合には、使用しているプロダクトの資料で、このステップを実行する方法を調べてください。
2. 使用する DB2 サーバーを選択して、InitializeServer() を呼び出し、データベースへの接続を初期化します。

   注: DB2 がユーザー ID とパスワードの妥当性検査を行うまでは、サーバーを初期化することはできません。 QMF (Windows 版) にユーザー ID とパスワードを要求するプロンプトを出させるか、または、ユーザー・アプリケーションにそれらを要求するプロンプトを出させ、InitializeServer() 関数呼び出しでパラメーターとしてそれらを渡すことができます。
3. InitializeQuery() を使用して、実行する照会を選択します。照会に変数が含まれている場合には、SetVariable() 関数を使用して、その変数の値を設定します。
4. 照会を開く、つまり実行します。 Open() 関数を使用して、SELECT ステートメントに対する照会のカーソルを開き、Execute() 関数を使用して、SELECT 以外のステートメントに対する SQL を実行します。

5. 照会が SELECT ステートメントである場合は、FetchNextRow() を繰り返し呼び出すことで、データの行を取り出します。一度に複数の行を取り出すには、FetchNextRows() または CompleteQuery() を使用して、QMF (Windows 版) にすべての行の取り出しを指示します。
6. 照会が SELECT ステートメントである場合は、Close() 関数を使用して照会を閉じます。
7. Commit() 関数または Rollback() 関数を使用して、作業単位を終了します。

呼び出しのブロック

QMF (Windows 版) API 関数は、すべて同期化されています。つまり、これらは、要求されたデータベース・アクションが完了するまでは、ブロックされています (つまり戻りません)。これは、クライアント・アプリケーションにおけるプログラミングを単純化するので、好ましいやり方と言えます。ただし、クライアント・アプリケーションが単一スレッドである場合は、QMF (Windows 版) API 関数の戻りを待っている間に、ユーザー入力に応答したり、画面の最新表示を行うことはできません。

QMF (Windows 版) API は、一度に 1 つのクライアントからの関数呼び出しに応答します。クライアント・アプリケーションがマルチスレッドである場合には、以下を実行することが必要です。

- ある 1 つの関数呼び出しが完了するのを待ってから、別の関数呼び出しを実行する。
- QMF (Windows 版) API の複数インスタンス (API を使用する各スレッドごとに 1 つ) を作成する。

データベースへの接続

QMF (Windows 版) API オブジェクトの各インスタンスは、照会を開くこと、データの取り出し、および SQL ステートメントの実行を含む、その後のロールバックまたはコミットを必要とするデータベースのすべてのアクティビティーのために、データベースへの単一の接続を作成し、使用します。

InitializeQuery() を 2 回以上呼び出すことによって、QMF (Windows 版) API オブジェクトの所定のインスタンスで複数の照会を作成した場合、その照会はすべて、同一の単一接続を共用します。

QMF (Windows 版) API は、一度に 1 つのクライアントからの関数呼び出しに応答します。ユーザーのクライアント・アプリケーションがマルチスレッドである場合には、以下を実行することが必要です。

- DeleteQMFObject()

- GetQMFObjectInfo()
- GetQMFObjectInfoEx()
- GetQMFObjectList()
- GetQMFObjectListEx()
- GetQMFQueryText()
- SaveQMFQuery()

QMF (Windows 版) は、管理的なデータベース・アクティビティー (たとえば、QMF 情報の検索) を処理するために、データベースへの 2 番目の接続を作成し、使用します。この 2 番目の接続は、クライアント・アプリケーションに対する一貫性のあるロールバックおよびコミットをサポートするために必要です。

QMF (Windows 版) API オブジェクトは、データベースに対するこれらの接続を自動的に処理します。ただし、システム管理者が、許可される接続の数に制限を設定してある場合には、QMF (Windows 版) API オブジェクトの各インスタンスが 2 つの接続を使用する場合があることに留意してください。

API についての解説

この解説では、QMF (Windows 版) API を使用したアプリケーション作成のために使用できるすべてのコマンドを列挙しています。

AddDecimalHostVariable()

short AddDecimalHostVariable(long *QueryID* , short *Type* , short *Precision* , short *Scale* , const VARIANT& *Value*)

説明

この関数は、*QueryID* で初期化された静的 SQL ステートメントの変数に、Value 内のデータを適用します。ステートメント内の各変数ごとに、この関数を呼び出します。 QMF (Windows 版) では、値と変数を突き合わせることはしないので、SQL ステートメント内の変数と同じ順序でこの関数を呼び出すことが必要です。

パラメーター

名前	説明
QueryID	InitializeStaticQuery() から戻される照会の ID。

Type	データベース・サーバーに渡される値の SQL データ・タイプ。この値は、VARIANT データ・タイプから、実際に渡される値への *Value* の変換に影響を及ぼします。 AddDecimalHostVariable() に対して有効な唯一の値は 484 (RSDT_DECIMAL) です。
Precision	10 進数値の精度。
Scale	10 進数値の位取り。
Value	ステートメントで置換するデータ値。ヌル値を指定する場合は、変形のタイプを VT_EMPTY に設定する必要があります。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

AddHostVariable()

short AddHostVariable(long *QueryID* , short *Type* , const VARIANT& *Value*)

説明

この関数は、*QueryID* で初期化された静的 SQL ステートメントの変数に、Value 内のデータを適用します。ステートメント内の各変数ごとに、この関数を呼び出すことが必要です。 QMF (Windows 版) では、値と変数を突き合わせることはしないので、SQL ステートメント内の変数と同じ順序でこの関数を呼び出すことが必要です。

パラメーター

名前	説明
QueryID	InitializeStaticQuery() から戻される照会の ID。
Type	データベース・サーバーに渡される値の SQL データ・タイプ。この値は、VARIANT データ・タイプから、実際に渡される値への *Value* の変換に影響を及ぼします。
Value	ステートメントで置換するデータ値。ヌル値を指定する場合は、変形のタイプを VT_EMPTY に設定する必要があります。

Type に有効な値には次のものがあります。

値	意味
384 (RSDT_DATE)	日付
388 (RSDT_TIME)	時刻
392 (RSDT_TIMESTAMP)	タイム・スタンプ

448 (RSDT_VARCHAR)	可変長文字ストリング
452 (RSDT_CHAR)	文字ストリング
464 (RSDT_VARGRAPHIC)	可変長グラフィック
468 (RSDT_GRAPHIC)	グラフィック
480 (RSDT_FLOAT)	浮動小数点数
496 (RSDT_INTEGER)	4 バイト整数
500 (RSDT_SMALLINT)	2 バイト整数

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

BindDecimalHostVariable()

short BindDecimalHostVariable(BSTR *CollectionName* , BSTR *PackageName* , short *SectionNumber* , short *Number* , BSTR *Name* , short *DataType* , short *Precision* , short *Scale*)

説明

この関数は、指定されたセクション内の変数をバインドします。テキスト ″:H″ を、ホスト変数のプレースホルダーとして SQL テキスト内に組み込みます。SQL テキスト内の 10 進ホスト変数のそれぞれに対して、BindDecimalHostVariable() を呼び出し、変数に関する情報を指定することが必要です。

パラメーター

名前	説明
CollectionName	バインドするパッケージの集合 ID。
PackageName	バインドするパッケージの名前。
SectionNumber	バインドする集合およびパッケージ内のステートメントのセクション番号。
Number	バインドする変数の ID。 SQL ステートメント内の最初の変数は、変数 0 というようになります。
Name	データベース・サーバーが診断目的で使用します。この値は、QMF (Windows 版) では、妥当性検査は行われず、また必要ともされません。
DataType	変数の SQL データ・タイプ。 BindDecimalHostVariable() に対して有効な唯一の値は 484 (RSDT_DECIMAL) です。

Precision	10 進数値の精度。
Scale	10 進数値の位取り。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

BindHostVariable()

short BindHostVariable(BSTR *CollectionName* , BSTR *PackageName* , short *SectionNumber* , short *Number* , BSTR *Name* , short *DataType* , short *Length*)

説明

この関数は、指定されたセクション内の変数をバインドします。テキスト ″:H″ を、ホスト変数のプレースホルダーとして SQL テキスト内に組み込みます。SQL テキスト内のホスト変数のそれぞれに対して、BindHostVariable() を呼び出し、変数に関する情報を指定することが必要です。

パラメーター

名前	説明
CollectionName	バインドするパッケージの集合 ID。
PackageName	バインドするパッケージの名前。
SectionNumber	バインドする集合およびパッケージ内のステートメントのセクション番号。
Number	バインドする変数の ID。 SQL ステートメント内の最初の変数は、変数 0 というようになります。
Name	データベース・サーバーが診断目的で使用します。この値は、QMF (Windows 版) では、妥当性検査は行われず、また必要ともされません。
DataType	変数の SQL データ・タイプ。
Length	変数の長さ。

DataType に有効な値には、次のものがあります。

値	意味
384 (RSDT_DATE)	日付
388 (RSDT_TIME)	時刻
392 (RSDT_TIMESTAMP)	タイム・スタンプ

448 (RSDT_VARCHAR)	可変長文字ストリング
452 (RSDT_CHAR)	文字ストリング
464 (RSDT_VARGRAPHIC)	可変長グラフィック
468 (RSDT_GRAPHIC)	グラフィック
480 (RSDT_FLOAT)	浮動小数点数
484 (RSDT_DECIMAL)	10 進数
496 (RSDT_INTEGER)	4 バイト整数
500 (RSDT_SMALLINT)	2 バイト整数

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

BindSection()

short BindSection(BSTR *CollectionName* , BSTR *PackageName* , short *SectionNumber* , BSTR *SQLText*)

説明

この関数は、バインディング時に、指定された集合およびパッケージのセクション番号で使用される SQL テキストを設定します。

パラメーター

名前	説明
CollectionName	バインドするパッケージの集合 ID。
PackageName	バインドするパッケージの名前。
SectionNumber	バインドする集合およびパッケージ内のステートメントのセクション番号。
SQLText	バインドするステートメントの SQL テキスト。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

CancelBind()

short CancelBind(BSTR *CollectionName* , BSTR *PackageName*)

説明

この関数は、以前に初期化されたバインド操作を取り消します。名前が指定されたパッケージに関するすべての情報が解放されます。

パラメーター

名前	説明
CollectionName	バインドするパッケージの集合 ID。
PackageName	バインドするパッケージの名前。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

ChangePassword()

short ChangePassword(BSTR *NewPassword*)

説明

この関数は、以前に InitializeServer() 呼び出しで指定されたユーザー ID に対するパスワードを変更します。

注: すべてのタイプのデータベース・サーバーでパスワード変更がサポートされるわけではありません。 InitializeServer() 呼び出しで指定されたサーバーがパスワード変更をサポートしていない場合は、エラーが戻され、パスワードは変更されません。

パラメーター

名前	説明
NewPassword	新規パスワード。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

ClearList()

short ClearList(short *Type*)

説明

この関数は、*Type* パラメーターで指定された内部リストを再度初期化します。

パラメーター

名前	説明
Type	RSL_SERVER または RSL_QUERY のいずれかの値。

戻り値

正常終了の場合はゼロ、異常終了の場合は RS_ERROR_OUTOFRANGE。

関連トピック

Open()

Close()

short Close(long *QueryID*)

説明

この関数は、照会を閉じて、*QueryID* を無効にします。照会用に開いているカーソルがある場合、そのカーソルが閉じられ、データベースは他のユーザー用に解放されます。この関数が、データベース・サーバーへの接続を終了させることはありません。接続は開いたままになるため、ロールバックまたはコミットは実行されません。

注: この関数の名前は、Microsoft Access 2.0 のキーワード Close と競合します。 Microsoft Access 2.0 を使用する場合には、関数名を大括弧 [ ] で囲んでください。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

Execute()
Open()

Commit()

short Close(long *QueryID*)

説明

この関数は、現行の作業単位で行った変更をコミットし、現行の作業単位を終了し、 開いているカーソルを閉じ、未解決のすべての照会 ID を無効にします。

注: この関数の名前は、Microsoft Access 2.0 のキーワード Commit と競合します。 Microsoft Access 2.0 を使用する場合には、関数名を大括弧 [ ] で囲んでください。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

Rollback()

CompleteQuery()

short CompleteQuery(long *QueryID*)

説明

この関数は、結果セットのすべての行を取り出して、それらを QMF (Windows 版) に内部的に保管します。照会用に開いているカーソルがある場合、そのカーソルが閉じられ、データベースは他のユーザー用に解放されます。
FetchNextRow() または FetchNextRows() を使用して、行を検索することができます。この照会が終了したら、Close() を呼び出します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

CopyToClipboard()

short CopyToClipboard(long *QueryID* , long *FirstRow* , long *FirstCol* , long *LastRow* , long *LastCol* , BOOL *IncludeColHeadings* , [VARIANT *DateTimeFomat*])

説明

この関数は、指定された範囲の行および列をクリップボードにコピーします。クリップボードにコピーしたいすべての行の行データをまだ検索していない場合は、この関数を呼び出す前に CompleteQuery() を呼び出します。データベースからまだ検索されていない行をコピーしようとすると、エラー・メッセージが戻されます。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
FirstRow	コピーに組み込む先頭の行。
FirstCol	コピーに組み込む先頭の列。
LastRow	コピーに組み込む最後の行、あるいは、すべての行を組み込む場合には -1。
LastCol	コピーに組み込む最後の列、あるいは、すべての列を組み込む場合には -1。
IncludeColHeadings	先頭行に列ヘッダーを組み込む場合はゼロ以外の値、列ヘッダーを組み込まない場合にはゼロ。
DateTimeFormat	日時の値に使用するフォーマット (オプショナル)。有効な値は 0 (ISO フォーマット)、1 (USA フォーマット)、2 (EUR フォーマット)、3 (JIS フォーマット)、4 (Windows コントロール・パネル・フォーマット) です。デフォルト値は 4 です。

注: 結果セット内の先頭行の値は 0、最後の行の値は合計の行数より 1 小さい値です。結果セット内の先頭列の値は 0、最後の列の値は合計の列数より 1 小さい値です。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。結果セットが空の場合、あるいはデータベースから行が検索されていない場合には、*FirstRow* =0 および *LastRow* =1 でない限り、ゼロ以外が戻されます。 FirstRow=0 で LastRow=1 の場合には、ゼロが戻され、空のストリングがクリップボードにコピーされます。

DeleteQMFObject()

short DeleteQMFObject(BSTR *OwnerAndName*)

説明

この関数は、QMF オブジェクト (照会、書式、プロシージャー、あるいは表) を削除します。

パラメーター

名前	説明
OwnerAndName	削除するオブジェクトの所有者と名前が、ピリオドで区切られて入っているストリング。例を以下に示します。 `John.Query2`

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

EndBind()

short EndBind(BSTR *CollectionName* , BSTR *PackageName*)

説明

この関数は、静的 SQL パッケージに対するバインド処理を完了します。この関数を呼び出すと、QMF (Windows 版) は、現行パッケージ用の完了情報を、処理のためにデータベースに送信します。

パラメーター

名前	説明
CollectionName	以前の StartBind() 呼び出しで使用された集合名。
PackageName	以前の StartBind() 呼び出しで使用されたパッケージ名。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

Execute()

short Execute(long *QueryID*)

説明

この関数は、SELECT 以外の SQL verb を使用する SQL ステートメントを実行します。ステートメントが何の結果も戻さない場合には Execute() を使用してください。結果を戻すステートメントの場合には、ExecuteEx() を使用してください。 SELECT verb を使用するステートメントの場合には、Execute() や ExecuteEx() ではなく、Open() を使用してください。照会が使用する verb を判別するには、GetQueryVerb() を呼び出します。

注: この関数の名前は、Microsoft Access 2.0 のキーワード Execute と競合します。 Microsoft Access 2.0 を使用する場合には、関数名を大括弧 [ ] で囲んでください。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

Execute()
Open()

ExecuteEx()

short ExecuteEx(long *QueryID,* VARIANT* *Result*)

説明

この関数は、SELECT 以外の SQL verb を使用する SQL ステートメントを実行します。ステートメントが結果を戻す場合、たとえば、SELECT INTO ステートメントでは、ExecuteEx() を使用してください。結果を戻さないステートメントの場合には、Execute() を使用してください。 SELECT verb を使用するステートメントの場合には、Execute() や ExecuteEx() ではなく、Open() を使用してください。照会が使用する verb を判別するには、GetQueryVerb() を呼び出します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Result	結果が保管される VARIANT のポインター。結果は、結果の各列ごとに 1 つの値が入っている配列 (変形タイプ VT_ARRAY | VT_VARIANT) となります。 各値は、そのネイティブ・データ・タイプ、あるいは最も近い変形データ・タイプのいずれかで指定されます。サポートされる戻りタイプは、string (変形タイプ VT_BSTR)、float (変形タイプ VT_R4)、 double (変形タイプ VT_R8)、short (変形タイプ VT_I2)、long (変形タイプ VT_I4)、および binary (変形タイプ VT_UI1 | VT_ARRAY) です。 この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

ExecuteStored Procedure()

short ExecuteStoredProcedure(long *QueryID,* [VARIANT *vaCommitOK*], [VARIANT *vaMaxResultSets*], [VARIANT *vaColumnNames*], [VARIANT *vaColumnLabels*], [VARIANT *vaColumnComments*])

説明

この関数は、SQL verb の CALL を使用する SQL ステートメントを実行して、データベース・サーバーでストアード・プロシージャー実行します。ストアード・プロシージャーが、(結果セットの代わり、または結果セットのほかに) 結果を何も戻さない場合に、ExecuteStoredProcedure() を使用します。結果を戻すストアード・プロシージャーの場合には、ExecuteStoredProcedureEx() を使用してください。

ExecuteStoredProcedure() を用いて実行するためにストアード・プロシージャーを初期化するには、まず最初に、CALL ステートメントを使用する SQL ステートメントを指定して InitializeQuery() を呼び出します。ストアード・プロシージャーの名前が、CALL ステートメントでリテラルとして指定されていることが必要です。 CALL ステートメントで指定されたパラメーター (定数またはそれ以外) はいずれも無視されます。その代わりに、AddHostVariable() を使用して、入力および出力の変数を指定します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。照会の SQL テキストは、CALL ステートメントを指定する必要があります。
vaCommitOK	ストアード・プロシージャーが作業単位をコミットすることができるか、あるいはこの操作が制限されるかどうかを指定するブール値 (オプショナル)。デフォルト値は True です。
vaMaxResultSets	ストアード・プロシージャーが戻すことのできる結果セットの最大数を指定する数値 (オプショナル)。ストアード・プロシージャーが結果セットを戻さないようにする場合、あるいはストアード・プロシージャーから DRDA を介して結果セットが戻されることをデータベース・サーバーがサポートしない場合には、ゼロを指定します。
vaColumnNames	戻される各結果セット内の列について、データベースが列名を戻す必要があるか否かを指定するブール値 (オプショナル)。
vaColumnLabels	戻される各結果セット内の列について、データベースが列ラベルを戻す必要があるか否かを指定するブール値 (オプショナル)。
vaColumnComments	戻される各結果セット内の列について、データベースが列コメントを戻す必要があるか否かを指定するブール値 (オプショナル)。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

ExecuteStored ProcedureEx()

short ExecuteStoredProcedureEx(long *QueryID,* VARIANT* *Result* , [VARIANT *vaCommitOK*], [VARIANT *vaMaxResultSets*], [VARIANT *vaColumnNames*], [VARIANT *vaColumnLabels*], [VARIANT *vaColumnComments*])

説明

この関数は、SQL verb の CALL を使用する SQL ステートメントを実行して、データベース・サーバーでストアード・プロシージャーを実行します。ExecuteStoredProcedureEx() は、ストアード・プロシージャーが、(結果セットの代わり、または結果セットのほかに) 結果を戻す場合に使用します。結果を戻すストアード・プロシージャーの場合には、ExecuteStoredProcedureEx() を使用してください。

ExecuteStoredProcedure() を用いて実行するためにストアード・プロシージャーを初期化するには、まず最初に、CALL ステートメントを使用する SQL ステートメントを指定して InitializeQuery() を呼び出します。ストアード・プロシージャーの名前が、CALL ステートメントでリテラルとして指定されていることが必要です。 CALL ステートメントで指定されたパラメーター (定数またはそれ以外) はいずれも無視されます。その代わりに、AddHostVariable() を使用して、入力および出力の変数を指定します。

ストアード・プロシージャーが結果セットを戻す場合は、GetStoredProcedureResultSets() を呼び出して、その結果セットの照会 ID を検索します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。照会の SQL テキストは、CALL ステートメントを指定する必要があります。

Result	結果が保管される VARIANT のポインター。結果は、結果の各列ごとに 1 つの値が入っている配列 (変形タイプ VT_ARRAY | VT_VARIANT) となります。 各値は、そのネイティブ・データ・タイプ、あるいは最も近い変形データ・タイプのいずれかで指定されます。サポートされる戻りタイプは、string (変形タイプ VT_BSTR)、float (変形タイプ VT_R4)、 double (変形タイプ VT_R8)、short (変形タイプ VT_I2)、long (変形タイプ VT_I4)、および binary (変形タイプ VT_UI1 | VT_ARRAY) です。 この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。
vaCommitOK	ストアード・プロシージャーが作業単位をコミットすることができるか、あるいはこの操作が制限されるかどうかを指定するブール値 (オプショナル)。デフォルトは True です。
vaMaxResultSets	ストアード・プロシージャーが戻すことのできる結果セットの最大数を指定する数値 (オプショナル)。ストアード・プロシージャーが結果セットを戻さないようにする場合、あるいはストアード・プロシージャーから DRDA を介して結果セットが戻されることをデータベース・サーバーがサポートしない場合には、ゼロを指定します。
vaColumnNames	戻される各結果セット内の列について、データベースが列名を戻す必要があるか否かを指定するブール値 (オプショナル)。
vaColumnLabels	戻される各結果セット内の列について、データベースが列ラベルを戻す必要があるか否かを指定するブール値 (オプショナル)。
vaColumnComments	戻される各結果セット内の列について、データベースが列コメントを戻す必要があるか否かを指定するブール値 (オプショナル)。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

Export()

short Export(long *QueryID* , long *FirstRow* , long *FirstCol* , long *LastRow* , long *LastCol* , short *Format* , short *StringDelimiter* , short *ColumnDelimiter* , BOOL *IncludeColHeadings* , BSTR *FileName* , [VARIANT *DateTimeFomat*])

説明

この関数は、指定された範囲の行および列をクリップボードにコピーします。クリップボードにコピーしたいすべての行の行データをまだ検索していない場合は、この関数を呼び出す前に CompleteQuery() を呼び出します。データベースからまだ検索されていない行をコピーしようとすると、エラー・メッセージが戻されます。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
FirstRow	エクスポートに組み込む先頭の行。
FirstCol	エクスポートに組み込む先頭の列。
LastRow	コピーに組み込む最後の行、あるいは、すべての行を組み込む場合には -1。
LastCol	コピーに組み込む最後の列、あるいは、すべての列を組み込む場合には -1。
IncludeColHeadings	先頭行に列ヘッダーを組み込む場合はゼロ以外の値、列ヘッダーを組み込まない場合にはゼロ。
Filename	エクスポートを書き込む先のファイルの名前が入っているストリング。
DateTimeFormat	日時の値に使用するフォーマット (オプショナル)。有効な値は 0 (ISO フォーマット)、1 (USA フォーマット)、2 (EUR フォーマット)、3 (JIS フォーマット)、4 (Windows コントロール・パネル・フォーマット) です。デフォルト値は 4 です。

注: 結果セット内の先頭行の値は 0、最後の行の値は合計の行数より 1 小さい値です。結果セット内の先頭列の値は 0、最後の列の値は合計の列数より 1 小さい値です。

名前	説明
Format	出力フォーマットを指定します。

値	意味
0 (RSEF_TEXT)	出力ファイルは、平文のテキスト・フォーマットで書かれます。
1 (RSEF_HTML)	出力ファイルは HTML フォーマットで書かれ、データは HTML 表に編成されます。
2 (RSEF_CSV)	出力ファイルは、CSV (コンマで区切った値) フォーマットで書かれます。

3 (RSEF_PCIXF)	出力ファイルは、PC/IXF フォーマットで書かれます。
4 (RSEF_S370IXF)	出力ファイルは、System/370 IXF フォーマットで書かれます。

名前	説明
String Delimiter	ストリング区切り文字を指定します。 *Format* が RSEF_HTML の場合には、このパラメーターは無視されます。

値	意味
0 (RSSD_NONE)	ストリング区切り文字は使用されません。
1 (RSSD_SINGLEQUOTE)	ストリングは、単一引用符文字 (') で区切られます。
2 (RSSD_DOUBLEQUOTE)	ストリングは、二重引用符文字 (") で区切られます。

名前	説明
Column Delimiter	列区切り文字を指定します。 *Format* が RSEF_HTML の場合には、このパラメーターは無視されます。

値	意味
0 (RSCD_SPACE)	列は、スペース文字 ( ) で区切られます。
1 (RSCD_TAB)	列は、タブ文字 (¥t) で区切られます。
2 (RSCD_COMMA)	列は、コンマ文字 (,) で区切られます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。結果セットが空の場合、あるいはデータベースから行が検索されていない場合には、*FirstRow* =0 および *LastRow* =1 でない限り、ゼロ以外が戻されます。FirstRow=0 で LastRow=1 の場合には、ゼロが戻され、空のファイルが書き込まれます。

関連トピック

CopyToClipboard()

ExportForm()

short ExportForm(BSTR *OwnerAndName* , BSTR *FileName*)

説明

この関数は、指定された QMF 書式を指定されたファイルにエクスポートします。

パラメーター

名前	説明
OwnerAndName	エクスポートしたい書式の所有者と名前が、ピリオドで区切られて入っているストリング。例を以下に示します。 `John.Query2`
FileName	エクスポートされた書式を書き込む先のファイルの名前が入っているストリング。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

PrintReport()

ExportReport()

short ExportReport(long *QueryID* , short *SourceType* , BSTR *Source* , BSTR *OutputFileName* , short *PageLength* , short *PageWidth* , BOOL *IncludeDateTime* , BOOL *IncludePageNumbers* , [VARIANT *Format*], [VARIANT *UseFormPageSetup*])

説明

この関数は、指定された照会についての報告書を作成して、それをファイルに書き込みます。 QMF 書式の報告書の書式設定とレイアウトを指定します。出力ファイルは、ASCII テキスト・ファイルで、各行はキャリッジ・リターン文字と改行文字の対で分離され、各ページは用紙送り文字で分離されます。出力ファイルを表示するには、固定ピッチ・フォントを使用するのが最適です。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Source	使用する書式の名前 (filename または Owner.Name)。
OutputFileName	報告書を書き込む先のファイルの名前。
PageLength	報告書の各ページの行数。 -1 という *PageLength* は、連続出力 (報告書の幅が *PageWidth* より広くない限り、ページの切れ目がない) を指定します。

IncludeDateTime	ゼロ以外の値は、日時が各ページの下部に組み込まれることを指定します。ゼロは、日時が組み込まれないことを指定します。
IncludePageNumbers	ゼロ以外の値は、ページ番号が各ページの下部に組み込まれることを指定します。ゼロは、ページ番号が組み込まれないことを指定します。
Format	エクスポートされる報告書の書式を指定します (オプショナル)。ゼロの場合、書式は平文のテキストであり、出力がその書式で作成されたものとまったく同じ (書式のタイプに応じて、テキストまたは HTML) でなければならないことを指定します。ゼロ以外の場合、書式は HTML で、出力は HTML でなければならないことを指定します。 HTML 以外の書式では、出力の先頭と末尾に HTML タグを追加することにより、出力が HTML に変換されます。デフォルト値はゼロです。
DateTimeFormat	日時の値に使用するフォーマット (オプショナル)。有効な値は 0 (ISO フォーマット)、1 (USA フォーマット)、2 (EUR フォーマット)、3 (JIS フォーマット)、4 (Windows コントロール・パネル・フォーマット) です。デフォルト値は 4 です。
Format	出力ファイルのフォーマット。
UseFormPageSetup	指定はオプショナルであり、ゼロ以外を指定すると、*PageLength* 、*PageWidth* 、*IncludeDateTime* 、および *IncludePageNumbers* の各パラメーターは無視され、その代わりにそれらの値が、指定された書式で保管されている値の中から採用されるようになります。デフォルト値はゼロです。

値	意味
0 (RSF_DEFAULT)	デフォルト書式を使用します。 *FormName* を空ストリングにする必要があります。
1 (RSF_DATABASE)	データベースからの書式を使用します。書式の所有者と名前 (Owner.Name) を *FormName* パラメーターで指定します。異なるデータベース・サーバーにある書式を使用する場合は、まず最初に ExportForm() を使用してその書式をファイルにエクスポートしてから、*SourceType* に RSF_FILE を指定します。
2 (RSF_FILE)	ファイルに入っている書式を使用します。そのファイル名を *FormName* パラメーターで指定します。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

ExportForm()

FastSaveData()

short FastSaveData(long *QueryID* , BOOL *Replace* , BSTR *Tablename* , BSTR *TableSpaceName* , [VARIANT *Comment*])

説明

この関数は、指定された照会についての報告書を作成して、それをファイルに書き込みます。 QMF 書式の報告書の書式設定とレイアウトを指定します。出力ファイルは、ASCII テキスト・ファイルで、各行はキャリッジ・リターン文字と改行文字の対で分離され、各ページは用紙送り文字で分離されます。出力ファイルを表示するには、固定ピッチ・フォントを使用するのが最適です。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Replace	指定されたデータを表の中の既存のデータと置換したい場合は、ゼロ以外の値を使用します。指定されたデータを、表の中の既存のデータに追加したい場合は、ゼロを使用します。
TableName	データを保管する表の名前。その表が存在しない場合、QMF (Windows 版) はその表を作成します。
TableSpaceName	表が存在しているか、または表が作成される表スペースの名前。 *TableSpaceName* が NULL もしくは空ストリングの場合には、QMF (Windows 版) はデフォルトの表スペースを使用します。常にデフォルトの表スペースを使用するように QMF (Windows 版) を構成してある場合には、このパラメーターは無視されます。 GetResourceLimit() の説明の中の RSR_SDDIFFERENTTS を参照してください。
Comment	データを保管する表についてのコメントを指定するストリング (オプショナル)。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

FetchNextRow()

short FetchNextRow(long *QueryID* , VARIANT* *Row*)

説明

この関数は、データの次の行をデータベースから取り出します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Row	結果が保管される VARIANT のポインター。結果は、結果の各列ごとに 1 つの値が入っている配列 (変形タイプ VT_ARRAY | VT_VARIANT) となります。配列内の値の数を判別するには、GetColumnCount() を呼び出します。 各値は、そのネイティブのデータ・タイプ、あるいは最も近い変形データ・タイプのいずれかで指定されます。サポートされる戻りタイプは、string (変形タイプ VT_BSTR)、float (変形タイプ VT_R4)、 double (変形タイプ VT_R8)、short (変形タイプ VT_I2)、long (変形タイプ VT_I4)、および binary (変形タイプ VT_UI1 | VT_ARRAY) です。 結果セットの終わりに達した (これ以上、取り出す行がない) とき、もしくは結果セットが空である場合、結果セットは、配列ではなく、空 (変形タイプ VT_EMPTY) となります。 . この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

注: Microsoft Excel 7.0 および Microsoft Access 7.0 (および、Visual Basic for Applications を使用する他の 32 ビットの Microsoft プロダクトも可能性があります) におけるバグが原因で、QMF (Windows 版) から受け取った Variant 変数内のストリング・データが、ユニコード (OLE が使用) から ANSI (VBA が使用) に変換されない場合があります。この状態が起きると、ストリングの先頭文字だけが表示されます。この問題を取り除くに

は、その変数を使用する QMF (Windows 版) の関数を呼び出す前に、その変数を空ストリングと等しくなるように設定しておきます。

戻り値
正常終了の場合はゼロ、異常終了の場合はゼロ以外。結果セットの終わりに達した場合、戻り値は -1 です。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック
FetchNextRows()

FetchNextRowEx()

short FetchNextRowEx(long *QueryID*)

説明
この関数は、データの次の行をデータベースから取り出します。この関数は、VARIANT 配列をサポートしない環境 (Microsoft Access 2.0 など) で使用することができます。この関数を GetColumnValue() と一緒に使用して、現在行の各列のデータを検索します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。

戻り値
正常終了の場合はゼロ、異常終了の場合はゼロ以外。結果セットの終わりに達した場合、戻り値は -1 です。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック
FetchNextRowsEx()

FetchNextRows()

short FetchNextRows(long *QueryID* , VARIANT* *Row* s, long* *NumRows*)

説明
この関数は、データの次の *NumRows* をデータベースから取り出します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Row	結果が保管される VARIANT のポインター。結果は、各行の各列ごとに 1 つの値が入っている二次元の配列 (変形タイプ VT_ARRAY |VT_VARIANT) となります。配列内の列の数を判別するには、GetColumnCount() を呼び出します。結果セット内の、取り出されない行数が *NumRows* より少ない場合 (この場合、配列には余分な未使用の項目も入っています) であっても、配列の次元は [*NumRows*][*ColumnCount*] となります。 各値は、そのネイティブのデータ・タイプ、あるいは最も近い変形データ・タイプのいずれかで指定されます。 サポートされる戻りタイプは、string (変形タイプ VT_BSTR)、float (変形タイプ VT_R4)、 double (変形タイプ VT_R8)、short (変形タイプ VT_I2)、long (変形タイプ VT_I4)、および binary (変形タイプ VT_UI1 | VT_ARRAY) です。 結果セットの終わりに達した (これ以上、取り出す行がない) とき、もしくは結果セットが空である場合、結果セットは、配列ではなく、空 (変形タイプ VT_EMPTY) となります。 この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。
NumRows	取り出す行の番号が入っている long のポインター。結果セット内の、取り出されない行数が *NumRows* より少ない場合には、*NumRows* は、その結果に含まれる実際の行数にリセットされます。

注: Microsoft Excel 7.0 および Microsoft Access 7.0 (および、Visual Basic for Applications を使用する他の 32 ビットの Microsoft プロダクトも可能性があります) におけるバグが原因で、QMF (Windows 版) から受け取った Variant 変数内のストリング・データが、ユニコード (OLE が使用) から ANSI (VBA が使用) に変換されない場合があります。この状態が起きると、ストリングの先頭文字だけが表示されます。この問題を取り除くには、その変数を使用する QMF (Windows 版) の関数を呼び出す前に、その変数を空ストリングと等しくなるように設定しておきます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。結果セットの終わりに達した場合、戻り値は -1 です。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

FetchNextRow()

FetchNextRowsEx()

short FetchNextRowsEx(long *QueryID* , long* *NumRows*)

説明

この関数は、データの次の *NumRows* をデータベースから取り出します。この関数は、VARIANT 配列をサポートしない環境 (Microsoft Access 2.0 など) で使用することができます。この関数を GetColumnValueEx() と一緒に使用して、所定の行の各列のデータを検索します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
NumRows	取り出す行の番号が入っている long のポインター。結果セット内の、取り出されない行数が *NumRows* より少ない場合には、*NumRows* は、その結果に含まれる実際の行数にリセットされます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。結果セットの終わりに達した場合、戻り値は -1 です。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

FetchNextRowEx()

FlushQMFCache()

void FlushQMFCache()

説明
この関数は、その QMF 情報のキャッシュをフラッシュして、その内容を破棄するよう QMF (Windows 版) に通知します。 QMF (Windows 版) は、次に QMF 情報が必要になったときは、データベースからその情報を入手します。通常、QMF (Windows 版) は、データベースから入手した QMF 情報をキャッシュに入れて、データベースのトラフィックを削減し、パフォーマンスを高めています。戻される情報が最新のものであるようにするために、GetQMFObjectInfo()、GetQMFQueryText()、または GetQMFObjectList() を呼び出す前にこの関数を呼び出します。

戻り値
なし。

GetColumnCount()

long GetColumnCount(long *QueryID*)

説明
この関数は、結果セット内の列数を戻します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。

戻り値
正常終了した場合は、各行の列の数。異常終了の場合は 0 または -1。戻り値が 0 もしくは -1 の場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetColumnDataValue()

short GetColumnDataValue(long *QueryID* , long *Index*)

説明
この関数は、データの現在行についての *Index* で指定された列のデータ値を戻します。この関数を呼び出した後、戻された値について *Value* プロパティーを問い合わせることができます。この関数を FetchNextRowEx() とともに使用して、データの単一行内のデータにアクセスします。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Index	検索するデータの行の、ゼロを基にした索引。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetColumnHeader()

BSTR GetColumnHeader(long *QueryID* , long *Index* , short* *Result*)

説明

この関数は、索引 *Index* に関連した列ヘッダー (列名) を戻します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Index	検索するデータの行の、ゼロを基にした索引。
Result	正常終了の場合はゼロ、異常終了の場合はゼロ以外。 Result がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

注: 静的 SQL ステートメントには、列ヘッダーは使用できません。 InitializeStaticQuery() から戻された照会 ID について、GetColumnHeader は ″Coln″ という形のストリングを戻します。ここで ″n″ は列番号です。

戻り値

戻されるストリングは、*Index* パラメーターで指定される列名を表しています。

GetColumnHeaderEx()

short GetColumnHeaderEx(long *QueryID* , long *Index*)

説明

この関数は、索引 *Index* に関連した列ヘッダー (列名) を戻します。この関数を呼び出した後は、戻された値について *Value* プロパティーを問い合わせることができます。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Index	検索するデータの行の、ゼロを基にした索引。

注: 静的 SQL ステートメントには、列ヘッダーは使用できません。 InitializeStaticQuery() から戻された照会 ID について、GetColumnHeader は ″Coln″ という形のストリングを戻します。ここで ″n″ は列番号です。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロの場合は、列名を表すストリングについての *Value* プロパティーを照会します。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetColumnHeadings()

short GetColumnHeadings(long *QueryID* , VARIANT* *Headings*)

説明

この関数は、列ヘッダー (列名とも言う) を戻します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Headings	結果が保管される VARIANT のポインター。結果は、各列ヘッダーごとに 1 つのストリングが入っている string の配列 (変形タイプ VT_ARRAY|VT_BSTR) となります。 この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。 Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

注: Microsoft Excel 7.0 および Microsoft Access 7.0 (および、Visual Basic for Applications を使用する他の 32 ビットの Microsoft プロダクトも可能性があります) におけるバグが原因で、QMF (Windows 版) から受け取った Variant 変数内のストリング・データが、ユニコード (OLE が使用) から ANSI (VBA が使用) に変換されない場合があります。この状態が起きると、ストリングの先頭文字だけが表示されます。この問題を取り除くには、その変数を使用する QMF (Windows 版) の関数を呼び出す前に、その変数を空ストリングと等しくなるように設定しておきます。

注: 静的 SQL ステートメントには、列ヘッダーは使用できません。InitializeStaticQuery() から戻される照会 ID について、GetColumnHeadings はストリング ″Col1″、″Col2″ などを戻します。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetColumnValue()

short GetColumnValue(long *QueryID* , long *Index* , VARIANT* *Value*)

説明

この関数は、データの現在行についての *Index* で指定された列のデータ値を戻します。この関数を FetchNextRowEx() とともに使用して、データの単一行内のデータにアクセスします。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Index	検索するデータの行の、ゼロを基にした索引。
Value	結果を保管する VARIANT のポインター。結果は、その変形タイプに基づいたデータ値です。 この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

戻り値
正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetColumnValueEx()

short GetColumnValueEx(long *QueryID* , long *RowIndex* , long *ColIndex* , VARIANT* *Value*)

説明
この関数は、*RowIndex* で指定されたデータの行について、 *ColIndex* で指定された列のデータ値を戻します。この関数を FetchNextRowsEx() とともに使用して、データの単一行内のデータにアクセスします。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
RowIndex	検索する行の、ゼロを基にした索引。
ColIndex	検索する列の、ゼロを基にした索引。
Value	結果を保管する VARIANT のポインター。結果の変形を照会することによって、さらに処理を進めるためのデータ・タイプを検出することができます。 この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

戻り値
正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetDefaultServerName()

BSTR GetDefaultServerName()

説明
この関数は、デフォルトのサーバー名が入っているストリングを戻します。

戻り値
デフォルトのサーバー名を指定するストリング。

GetGlobalVariable()

BSTR GetGlobalVariable(BSTR *Name*)

説明
この関数は、指定されたグローバル変数の値を検索します。

パラメーター

名前	説明
Name	設定する変数の名前が入っているストリング。

戻り値
グローバル変数値が入っているストリング、もしくは、変数に値がない場合またはエラーが発生した場合は NULL。

GetHostVariableNames()

short GetHostVariableNames(long *QueryID* , VARIANT* *Names*)

説明
この関数は、指定された照会で参照されるすべてのホスト変数の名前の配列を戻します。照会は、ホスト変数 (QMF 照会を用いて保管されたか、または AddHostVariable() によって作成されたもの) を参照する静的照会でなければなりません。

パラメーター

Name	説明
QueryID	InitializeQuery() から戻される照会の ID。
Names	結果の配列を保管する VARIANT のポインター。

戻り値
正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合、GetLastErrorString() を呼び出すと、詳しいエラー情報を入手することができます。

GetHostVariableTypes()

short GetHostVariableTypes(long *QueryID* , VARIANT* *Types*)

説明

この関数は、指定された照会で参照されるすべてのホスト変数のデータ・タイプの配列を戻します。照会は、ホスト変数 (QMF 照会を用いて保管されたか、または AddHostVarialble() によって作成されたもの) を参照する静的照会でなければなりません。戻すことができるデータ・タイプのリストについては、AddHostVariable() を参照してください。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Types	結果の配列を保管する VARIANT のポインター。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合、GetLastErrorString() を呼び出すと、詳しいエラー情報を入手することができます。

GetLastErrorString()

BSTR GetLastErrorString()

説明

この関数は、最新のエラーに関する情報が入っているストリングを戻します。(エラーなしで) 正常に実行された関数の後にこの関数を呼び出すと、この関数は、前の関数呼び出し時に発生した最後のエラーに関する情報を戻します。混乱を避けるために、この関数は、必ず、エラーを戻した関数の呼び出し直後に呼び出すようにしてください。

戻り値

エラー情報が入っているストリング。 QMF API オブジェクトを作成してからエラーが発生していない場合には、NULL が戻されます。

関連トピック

GetLastErrorType()
GetLastSQLCode()
GetLastSQLError()
GetLastSQLState()

GetLastErrorType()

short GetLastErrorType()

説明

この関数は、最新のエラーのタイプを戻します。 (エラーなしで) 正常に実行された関数の後にこの関数を呼び出すと、この関数は、前の関数呼び出し時に発生した最後のエラーに関する情報を戻します。混乱を避けるために、この関数は、必ず、エラーを戻した関数の呼び出し直後に呼び出すようにしてください。

戻り値

戻された値は次のようにエラーのタイプを示しています。

値	意味
0 (RS_ERROR_NONE)	QMF (Windows 版) API オブジェクトが作成されてから、以後、エラーは発生していません。
1 (RS_ERROR_SQL)	SQL エラーが発生しました。引き数として *QueryID* を採用する関数の呼び出し時にエラーが発生した場合には、*Close()* を呼び出して、照会を閉じてください。ロールバックは実行されません。 QMF (Windows 版) API オブジェクトを使用し続けることはできますが、さらにエラーに遭遇する可能性があります。
2 (RS_ERROR_USER_CANCEL)	一般的には、使用中のウィンドウの「キャンセル」をクリックすることで、ユーザーが操作を取り消しました。これにより、QMF (Windows 版) は暗黙的にロールバックを実行 (未解決のすべての照会 ID を無効にして) し、データベースへの接続を破棄します。継続したい場合には、InitializeServer() もしくは ReinitializeServer() を呼び出すことが必要です。
3 (RS_ERROR_FATAL_GOV)	致命的な管理プログラム・エラーが発生しました。考えられることの 1 つとして、許可されている最大アイドル時間を超過したために、QMF (Windows 版) API がタイムアウトになったことが挙げられます。これにより、QMF (Windows 版) は暗黙的にロールバックを実行 (未解決のすべての照会 ID を無効にして) し、データベースへの接続を破棄します。継続したい場合には、InitializeServer() もしくは ReinitializeServer() を呼び出すことが必要です。

4 (RS_ERROR_NONFATAL_GOV)	致命的ではない管理プログラム・エラーが発生しました。取り出しを認められている最大の行数を超過したか、もしくは SQL verb が認められているものではありません。引き数として *QueryID* を採用する関数の呼び出し時にエラーが発生した場合には、Close() を呼び出して、その照会をクローズしてください。ロールバックは実行されず、またデータベースへの接続にも影響がないため、 QMF (Windows 版) API オブジェクトの使用を続行することができます。
5 (RS_ERROR_OTHER)	一般的なエラーが発生しました。ロールバックは実行されません。 QMF (Windows 版) API オブジェクトを使用し続けることはできますが、さらにエラーに遭遇する可能性があります。

関連トピック

GetLastErrorString()
GetLastSQLCode()
GetLastSQLError()
GetLastSQLState()

GetLastSQLCode()

long GetLastSQLCode()

説明

この関数は、最新のエラーの SQL コードを戻します。 (エラーなしで) 正常に実行された関数の後にこの関数を呼び出すと、この関数は、前の関数呼び出し時に発生した最後のエラーに関する情報を戻します。混乱を避けるために、この関数は、必ず、エラーを戻した関数の呼び出し直後に呼び出すようにしてください。

戻り値

最新のエラーの SQL コード。 QMF (Windows 版) API オブジェクトを作成してから、以後、エラーが発生していない場合、あるいは最新のエラーが SQL エラーではなかった場合は、ゼロが戻されます。

関連トピック

GetLastErrorString()
GetLastErrorType()
GetLastSQLError()
GetLastSQLState()

GetLastSQLError()

VARIANT GetLastSQLError()

説明

この関数は、最新のエラーの詳細なエラー情報を戻します。 (エラーなしで) 正常に実行された関数の後にこの関数を呼び出すと、この関数は、前の関数呼び出し時に発生した最後のエラーに関する情報を戻します。混乱を避けるために、この関数は、必ず、エラーを戻した関数の呼び出し直後に呼び出すようにしてください。

戻り値

エラー情報が入っている配列 (変形タイプ VT_ARRAY | VT_VARIANT)。 QMF (Windows 版) API オブジェクトを作成してから、以後、エラーが発生していない場合、あるいは最新のエラーが SQL エラーではなかった場合は、空 (変形タイプ VT_EMPTY) が戻されます。配列には、次のフォーマットがあります。

エレメント	タイプ	内容
0	long (VT_I4)	コード
1	string (VT_BSTR)	状態
2	string (VT_BSTR)	ErrProc
3	string (VT_BSTR)	RDBName
4	long (VT_I4)	ErrD1
5	long (VT_I4)	ErrD2
6	long (VT_I4)	ErrD3
7	long (VT_I4)	ErrD4
8	long (VT_I4)	ErrD5
9	long (VT_I4)	ErrD6
10	string (VT_BSTR)	Warn0
11	string (VT_BSTR)	Warn1
12	string (VT_BSTR)	Warn2
13	string (VT_BSTR)	Warn3
14	string (VT_BSTR)	Warn4
15	string (VT_BSTR)	Warn5
16	string (VT_BSTR)	Warn6
17	string (VT_BSTR)	Warn7
18	string (VT_BSTR)	Warn8

19	string (VT_BSTR)	Warn9
20	string (VT_BSTR)	WarnA
21	string (VT_BSTR)	MessageTokens

関連トピック

GetLastErrorString()
GetLastErrorType()
GetLastSQLCode()
GetLastSQLState()

GetLastSQLState()

BSTR GetLastSQLState()

説明

この関数は、最新のエラーの SQL 状態を戻します。 (エラーなしで) 正常に実行された関数の後にこの関数を呼び出すと、この関数は、前の関数呼び出し時に発生した最後のエラーに関する情報を戻します。混乱を避けるために、この関数は、必ず、エラーを戻した関数の呼び出し直後に呼び出すようにしてください。

戻り値

最新のエラーの SQL コードが入っているストリング。 QMF (Windows 版) API オブジェクトを作成してから、以後、エラーが発生していない場合、あるいは最新のエラーが SQL エラーではなかった場合は、NULL が戻されます。

関連トピック

GetLastErrorString()
GetLastErrorType()
GetLastSQLCode()
GetLastSQLError()

GetOption()

short GetOption(short *Option* , VARIANT* *Value*)

説明

QMF (Windows 版) で、指定されたオプション値を入手します。

パラメーター

名前	説明
Option	検索するオプションを指定します。

値	意味
0 (RSO_SERVER_DEFINITION_FILE)	サーバーの定義ファイル名
1 (RSO_CPIC_DLL)	CPI-C プロバイダーの DLL ファイル名
2 (RSO_CPIC_TIMEOUT_WARNING)	CPI-C 警告タイムアウト (秒数)。この制限は、QMF (Windows 版) API が使用するものではありません。
3 (RSO_CPIC_TIMEOUT_CANCEL)	CPI-C 取り消しタイムアウト (秒数)。
4 (RSO_TCP_TIMEOUT_WARNING)	TCP 警告タイムアウト (秒数)。この制限は、QMF (Windows 版) API が使用するものではありません。
5 (RSO_TCP_TIMEOUT_CANCEL)	TCP 取り消しタイムアウト (秒数)。
6 (RSO_DISPLAY_NULLS_STRING)	ヌル値を表示するために使用するストリング。
7 (RSO_ENTER_NULLS_STRING)	ヌル値を入力するために使用するストリング。
8 (RSO_ENTER_DEFAULTS_STRING)	デフォルト値を入力するために使用するストリング。
9 (RSO_TRACE_FILE_1)	トレース・ファイル 1 の名前。
10 (RSO_TRACE_FILE_2)	トレース・ファイル 2 の名前。
11 (RSO_TCP_TRACE_LEVEL)	TCP トレース・レベル。
12 (RSO_CPIC_TRACE_LEVEL)	CPI-C トレース・レベル。
13 (RSO_DDM_TRACE_LEVEL)	DDM トレース・レベル。

Value	結果が保管される VARIANT のポインター。結果は、結果の各列ごとに 1 つの値が入っている配列 (変形タイプ VT_ARRAY | VT_VARIANT) となります。配列内の値の数を判別するには、GetColumnCount() を呼び出します。この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。 Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

注: Microsoft Excel 7.0 および Microsoft Access 7.0 (および、Visual Basic for Applications を使用する他の 32 ビットの Microsoft プロダクトも可能性があります) におけるバグが原因で、QMF (Windows 版) から受け取った Variant 変数内のストリング・データが、ユニコード (OLE が使用) から ANSI (VBA が使用) に変換されない場合があります。この状態が起きると、ストリングの先頭文字だけが表示されます。この問題を取り除くに

は、その変数を使用する QMF (Windows 版) の関数を呼び出す前に、その変数を空ストリングと等しくなるように設定しておきます。

戻り値
正常終了の場合はゼロ、ゼロ以外の場合は異常終了。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック
SetOption()

GetOptionEx()

short GetOptionEx(short *Option*)

説明
QMF (Windows 版) で、指定されたオプション値を入手します。オプション値が戻されたら、そのデータの *Option* プロパティーを照会することが必要です。

パラメーター

名前	説明
Option	オプション値は、GetOption() 呼び出しの場合と同じです。

戻り値
正常終了の場合はゼロ、ゼロ以外の場合は異常終了。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック
GetOption()
SetOption()

GetProcText()

BSTR GetProcText(long *ProcID*)

説明
この関数は、変数置換後に、指定されたプロシージャーに対して実行されるテキストを戻します。この関数を呼び出す前に、SetProcVariable() を使用して、プロシージャーで使用する変数の値を設定しておくことが必要です。

パラメーター

名前	説明
ProcID	InitializeProc() から戻されるプロシージャーの ID。

戻り値

正常終了の場合は、プロシージャー・テキストが入っているストリングが戻されます。異常終了の場合は、NULL が戻されます。戻り値が NULL の場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetProcVariables()

short GetProcVariables(long *ProcID* , VARIANT* *Variables*)

説明

QMF (Windows 版) で、指定されたオプション値を入手します。

パラメーター

名前	説明
ProcID	InitializeProc() から戻されるプロシージャーの ID。
Value	結果が保管される VARIANT のポインター。結果は、string の配列 (変形タイプ VT_ARRAY | VT_BSTR) となり、各ストリングには 1 つの変数の名前が入っています。プロシージャーに変数がない場合には、結果は空 (変形タイプ VT_EMPTY) となります。この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

注: Microsoft Excel 7.0 および Microsoft Access 7.0 (および、Visual Basic for Applications を使用する他の 32 ビットの Microsoft プロダクトも可能性があります) におけるバグが原因で、QMF (Windows 版) から受け取った Variant 変数内のストリング・データが、ユニコード (OLE が使用) から ANSI (VBA が使用) に変換されない場合があります。この状態が起きると、ストリングの先頭文字だけが表示されます。この問題を取り除くには、その変数を使用する QMF (Windows 版) の関数を呼び出す前に、その変数を空ストリングと等しくなるように設定しておきます。

戻り値

正常終了の場合はゼロ、ゼロ以外の場合は異常終了。プロシージャーに変数がない場合、戻り値は RS_NO_ERROR_NO_DATA (-1) です。戻り値がゼロ以外

である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetQMFObjectInfo()

short GetQMFObjectInfo(BSTR *OwnerAndName* , short *Type* , short *Time* , VARIANT* *Value*)

説明

この関数は、QMF オブジェクト (書式または照会のいずれか) に関する情報を戻します。戻される情報は、*Type* および *Time* の各パラメーターで指定されます。

パラメーター

名前	説明
OwnerAndName	検索する情報の対象であるオブジェクトの所有者と名前が、ピリオドで区切られて入っているストリング。例を以下に示します。 `John.Query2`
Value	結果が保管される VARIANT のポインター。RSI_TIMEUSED、RSI_TIMESRUN、RSI_TIMESCANCELLED、および RSI_LEVEL の場合、結果は short (変形タイプ VT_I2) となります。RSI_RESTRICTED の場合、結果は boolean (変形タイプ VT_BOOL) となります。それ以外の場合はすべて、結果は string (変形タイプ VT_BSTR) となります。この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。 Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

注: Microsoft Excel 7.0 および Microsoft Access 7.0 (および、Visual Basic for Applications を使用する他の 32 ビットの Microsoft プロダクトも可能性があります) におけるバグが原因で、QMF (Windows 版) から受け取った Variant 変数内のストリング・データが、ユニコード (OLE が使用) から ANSI (VBA が使用) に変換されない場合があります。この状態が起きると、ストリングの先頭文字だけが表示されます。この問題を取り除くには、その変数を使用する QMF (Windows 版) の関数を呼び出す前に、その変数を空ストリングと等しくなるように設定しておきます。

Type	入手する情報のタイプを指定します。

値	意味
0 (RSI_COMMENT)	コメント
1 (RSI_LEVEL)	レベル
2 (RSI_TYPE)	タイプ
3 (RSI_SUBTYPE)	サブタイプ
4 (RSI_RESTRICTED)	制限付き
5 (RSI_MODEL)	モデル
6 (RSI_TIMESUSED)	使用された回数
7 (RSI_TIMESRUN)	実行された回数
8 (RSI_TIMESCANCELLED)	取り消された回数
9 (RSI_DATE)	最初に使用された日付、最後に使用された日付、または最後に変更された日付
10 (RSI_TIME)	最初に使用された時刻、最後に使用された時刻、または最後に変更された時刻
11 (RSI_USERID)	最初に使用されたユーザー ID、最後に使用されたユーザー ID、または最後に変更されたユーザー ID
12 (RSI_SQLID)	最初に使用された SQL ID、最後に使用された SQL ID、または最後に変更された SQL ID
13 (RSI_ENVIRONMENT)	最初に使用された環境、最後に使用された環境、または最後に変更された環境
14 (RSI_MODE)	最初に使用されたモード、最後に使用されたモード、または最後に変更されたモード
15 (RSI_COMMAND)	最初に使用されたコマンド、最後に使用されたコマンド、または最後に変更されたコマンド

Time	最初に使用されたか、最後に使用されたか、または最後に変更されたかを指定します。

値	意味
0 (RST_FIRSTUSED)	最初に使用された
1 (RST_LASTUSED)	最後に使用された
2 (RST_LASTMODIFIED)	最後に変更された

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetQMFObjectInfoEx()

short GetQMFObjectInfoEx(BSTR *OwnerAndName* , short *Type* , short *Time*)

説明

この関数は、QMF オブジェクトに関する情報を戻します。戻される情報は、*Type* および *Time* の各パラメーターで指定されます。この関数を呼び出した後は、戻された値について *QMFObjectInfo* プロパティーを問い合わせることができます。

パラメーター

名前	説明
OwnerAndName	検索する情報の対象であるオブジェクトの所有者と名前が、ピリオドで区切られて入っているストリング。例を以下に示します。 `John.Query2`
Type	獲得する情報のタイプを指定します。

値	意味
0 (RSI_COMMENT)	コメント
1 (RSI_LEVEL)	レベル
2 (RSI_TYPE)	タイプ
3 (RSI_SUBTYPE)	サブタイプ
4 (RSI_RESTRICTED)	制限付き
5 (RSI_MODEL)	モデル
6 (RSI_TIMESUSED)	使用された回数
7 (RSI_TIMESRUN)	実行された回数
8 (RSI_TIMESCANCELLED)	取り消された回数
9 (RSI_DATE)	最初に使用された日付、最後に使用された日付、または最後に変更された日付
10 (RSI_TIME)	最初に使用された時刻、最後に使用された時刻、または最後に変更された時刻

11 (RSI_USERID)	最初に使用されたユーザー ID、最後に使用されたユーザー ID、または最後に変更されたユーザー ID
12 (RSI_SQLID)	最初に使用された SQL ID、最後に使用された SQL ID、または最後に変更された SQL ID
13 (RSI_ENVIRONMENT)	最初に使用された環境、最後に使用された環境、または最後に変更された環境
14 (RSI_MODE)	最初に使用されたモード、最後に使用されたモード、または最後に変更されたモード
15 (RSI_COMMAND)	最初に使用されたコマンド、最後に使用されたコマンド、または最後に変更されたコマンド

Time	最初に使用されたか、最後に使用されたか、または最後に変更されたかを指定します。

値	意味
0 (RST_FIRSTUSED)	最初に使用された
1 (RST_LASTUSED)	最後に使用された
2 (RST_LASTMODIFIED)	最後に変更された

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetQMFObjectList()

short GetQMFObjectList(BSTR *Owner* , BSTR *Name* , short *Type* , VARIANT* *List*)

説明

この関数は、*Owner* および *Name* の両パラメーターで指定されたパターンと一致するすべての QMF オブジェクトの名前の配列を戻します。

パラメーター

名前	説明
Owner	戻されるリストに組み込むオブジェクトの所有者が入っているストリング。

Name	戻されるリストに組み込むオブジェクトの名前が入っているストリング。
List	結果が保管される VARIANT のポインター。結果は、string の配列 (変形タイプ VT_ARRAY | VT _BSTR) となり、それぞれのストリングのフォーマットは Owner.Name です。一致する QMF (Windows 版) 照会が検出されなかった場合、結果は空 (変形タイプ VT_EMPTY) となります。この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。 Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

注: Microsoft Excel 7.0 および Microsoft Access 7.0 (および、Visual Basic for Applications を使用する他の 32 ビットの Microsoft プロダクトも可能性があります) におけるバグが原因で、QMF (Windows 版) から受け取った Variant 変数内のストリング・データが、ユニコード (OLE が使用) から ANSI (VBA が使用) に変換されない場合があります。この状態が起きると、ストリングの先頭文字だけが表示されます。この問題を取り除くには、その変数を使用する QMF (Windows 版) の関数を呼び出す前に、その変数を空ストリングと等しくなるように設定しておきます。

Type	リストに組み込む QMF オブジェクトのタイプを指定します。下記の値を一緒に追加して、複数のオブジェクト・タイプを指定することができます。

値	意味
2048 (RSQ_MASK_QUERIES)	リストに QMF 照会を組み込みます。
1024 (RSQ_MASK_FORMS)	リストに QMF 書式を組み込みます。
512 (RSQ_MASK_PROCS)	リストに QMF プロシージャーを組み込みます。
256 (RSQ_MASK_TABLES)	リストに表を組み込みます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。一致する QMF オブジェクトが検出されなかった場合、戻り値はゼロです。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetQMFObjectListEx()

short GetQMFObjectListEx(BSTR *Owner* , BSTR *Name* , short *Index*)

説明

この関数は、*Index* パラメーターで参照される、*Owner* および *Name* の両パラメーターで指定されたパターンと一致する QMF オブジェクトの名前を戻します。この関数を呼び出した後は、戻された値について *Value* プロパティーを問い合わせることができます。

パラメーター

名前	説明
Owner	戻されるリストに組み込むオブジェクトの所有者が入っているストリング。
Name	戻されるリストに組み込むオブジェクトの名前が入っているストリング。
Index	パターンと一致する QMF オブジェクトのリストの索引。
Type	リストに組み込む QMF オブジェクトのタイプを指定します。下記の値を一緒に追加して、複数のオブジェクト・タイプを指定することができます。

値	意味
2048 (RSQ_MASK_QUERIES)	リストに QMF 照会を組み込みます。
1024 (RSQ_MASK_FORMS)	リストに QMF 書式を組み込みます。
512 (RSQ_MASK_PROCS)	リストに QMF プロシージャーを組み込みます。
256 (RSQ_MASK_TABLES)	リストに表を組み込みます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。一致する QMF オブジェクトが検出されなかった場合、戻り値は RS_ERROR_OUTOFRANGE となります。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetQMFProcText()

BSTR GetQMFProcText(BSTR *OwnerAndName*)

説明

この関数は、変数置換後に、指定されたプロシージャーに対して実行されるテキストを戻します。この関数を呼び出す前に、SetProcVariable() を使用して、プロシージャーで使用する変数の値を設定しておくことが必要です。

パラメーター

名前	説明
OwnerAndName	削除するオブジェクトの所有者と名前が、ピリオドで区切られて入っているストリング。例を以下に示します。 `John.Proc2`

戻り値

検索されたプロシージャーのためのテキストが入っているストリング、もしくは、プロシージャーを検索できなかった場合は NULL。戻り値が NULL の場合は、GetLastErrorString()、GetLastErrorType()、GetLastSQLCode()、GetLastSQLError、または GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetQMFQueryText()

BSTR GetQMFQueryText(BSTR *OwnerAndName*)

説明

この関数は、指定された照会で保管されている SQL テキストを検索します。

パラメーター

名前	説明
OwnerAndName	削除するオブジェクトの所有者と名前が、ピリオドで区切られて入っているストリング。例を以下に示します。 `John.Query2`

戻り値

検索された照会のためのテキストが入っているストリング、もしくは、照会を検索できなかった場合は NULL。戻り値が NULL の場合は、GetLastErrorString()、GetLastErrorType()、GetLastSQLCode()、GetLastSQLError、または GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetQueryText()

BSTR GetQueryText(long *QueryID*)

説明

この関数は、変数置換後に、指定された照会に対して実行される SQL テキストを戻します。この関数を呼び出す前に、SetVariable() を使用して、照会で使用される変数の値を設定しておくことが必要です。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。

注: 静的 SQL ステートメントには、照会テキストは使用できません。InitializeStaticQuery() から戻される照会 ID について、GetQueryText() は空ストリングを戻します。

戻り値

正常終了の場合は、SQL テキストが入っているストリングが戻されます。異常終了の場合は、NULL が戻されます。戻り値が NULL の場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetQueryVerb()

BSTR GetQueryVerb(long *QueryID*)

説明

この関数は、照会で使用された SQL verb が入っているストリングを戻します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。

注: 静的 SQL ステートメントには、照会 verb は使用できません。InitializeStaticQuery() から戻される照会 ID について、GetQueryVerb() は空ストリングを戻します。

戻り値

正常終了の場合は、SQL verb が入っているストリングが戻されます。異常終了の場合は、NULL が戻されます。戻り値が NULL の場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetResourceLimit()

short GetResourceLimit(short *Resource* , long* *Value*)

説明

この関数は、要求されたリソース限界を獲得します。リソース限界はサーバーごとに処理されるため、この関数を呼び出す前に、InitializeServer() を呼び出すことが必要です。

パラメーター

名前	説明
Resource	リソースの値には次のものがあります。

値	意味
0 (RSR_IDLE_CONNECTION_TIMEOUT)	アイドル接続タイムアウト (秒数)
1 (RSR_IDLE_QUERY_TIMEOUT_CANCEL)	アイドル照会タイムアウト (秒数)
2 (RSR_IDLE_QUERY_TIMEOUT_WARNING)	アイドル照会タイムアウト (秒数)。これは、警告限界です。QMF (Windows 版) API では強制されるものではありません。
3 (RSR_SERVER_RESPONSE_TIMEOUT_CANCEL)	サーバーのタイムアウト (秒数)。
4 (RSR_SERVER_RESPONSE_TIMEOUT_WARNING)	サーバーのタイムアウト (秒数)。これは、警告限界です。QMF (Windows 版) API では強制されるものではありません。
5 (RSR_MAX_ROWS_TO_FETCH_CANCEL)	取り出す行の最大数。
6 (RSR_MAX_ROWS_TO_FETCH_WARNING)	取り出す行の最大数。これは、警告限界です。QMF (Windows 版) API では強制されるものではありません。
7 (RSR_MAX_BYTES_TO_FETCH_CANCEL)	取り出すバイトの最大数。
8 (RSR_MAX_BYTES_TO_FETCH_WARNING)	取り出すバイトの最大数。これは、警告限界です。QMF (Windows 版) API では強制されるものではありません。

9 (RSR_MAX_CONNECTIONS)	データベース・サーバーに対して許可される接続の最大数
10 (RSR_ALLOW_SERVER_ACCESS_UI)	QMF (Windows 版) インターフェースからデータベース・サーバーへのアクセスが認められているか。
11 (RSR_ALLOW_SERVER_ACCESS_API)	QMF (Windows 版) API からデータベース・サーバーへのアクセスが認められているか。
12 (RSR_FETCH_ALL_ROWS)	すべての行を自動的に取り出すか。
13 (RSR_CONFIRM_UPDATES)	データベース・サーバーの更新を確認するか。このオプションは、QMF (Windows 版) API には影響を及ぼしません。データベースの更新は、QMF (Windows 版) API に対して確認されることはありません。
14 (RSR_SUMMARY_TRACKING)	要約オブジェクトのトラッキングは使用可能か。
15 (RSR_DETAILED_TRACKING)	詳細オブジェクトのトラッキングは使用可能か。
16 (RSR_SQL_TRACKING)	SQL テキストのトラッキングは使用可能か。
17 (RSR_ADHOC_TRACKING)	随時オブジェクトのトラッキングは使用可能か。
18 (RSR_ALLOW_ACQUIRE)	SQL verb の ACQUIRE は許可されているか。
19 (RSR_ALLOW_ALTER)	SQL verb の ALTER は許可されているか。
20 (RSR_ALLOW_COMMENT)	SQL verb の COMMENT は許可されているか。
21 (RSR_ALLOW_CREATE)	SQL verb の CREATE は許可されているか。
22 (RSR_ALLOW_DELETE)	SQL verb の DELETE は許可されているか。

23 (RSR_ALLOW_DROP)	SQL verb の DROP は許可されているか。
24 (RSR_ALLOW_EXPLAIN)	SQL verb の EXPLAIN は許可されているか。
25 (RSR_ALLOW_GRANT)	SQL verb の GRANT は許可されているか。
26 (RSR_ALLOW_INSERT)	SQL verb の INSERT は許可されているか。
27 (RSR_ALLOW_LABEL)	SQL verb の LABEL は許可されているか。
28 (RSR_ALLOW_LOCK)	SQL verb の LOCK は許可されているか。
29 (RSR_ALLOW_REVOKE)	SQL verb の REVOKE は許可されているか。
30 (RSR_ALLOW_SELECT)	SQL verb の SELECT は許可されているか。
31 (RSR_ALLOW_SET)	SQL verb の SET は許可されているか。
32 (RSR_ALLOW_SIGNAL)	SQL verb の SIGNAL は許可されているか。
33 (RSR_ALLOW_UPDATE)	SQL verb の UPDATE は許可されているか。
34 (RSR_ALLOW_CALL)	SQL verb の CALL は許可されているか。
35 (RSR_ALLOW_SAVE_DATA)	Save Data コマンドは許可されているか。
36 (RSR_SAVE_DATA_TABLE_SPACE_NAME)	パッケージをバインディングするためのデフォルト集合名は。
37 (RSR_SAVE_DATA_TABLE_SPACE_NAME_OVERRIDE)	Save Data コマンドに対するデフォルトの表スペース名をユーザーが変更できるか。
38 (RSR_ALLOW_BIND_PACKAGE)	パッケージのバインディングを許可するか。
39 (RSR_DEF_COLLECTION)	パッケージをバインディングするためのデフォルト集合名

40 (RSR_DEF_COLLECTION_OVERRIDE)	パッケージをバインディングするためのデフォルト集合名をユーザーが変更できるか。
41 (RSR_DEF_ISOLATION_LEVEL)	パッケージをバインディングするためのデフォルト分離レベル
42 (RSR_DEF_ISOLATION_LEVEL_OVERRIDE)	パッケージをバインディングするためのデフォルト分離レベルをユーザーが変更できるか。
43 (RSR_ALLOW_TABLE_EDIT)	表編集プログラムの使用を許可するか。
44 (RSR_ALLOW_EXPORT)	データのエクスポートを許可するか。
45 (RSR_ALLOW_SAVED_QUERIES_ONLY)	ユーザーに保管済み照会だけの実行を許可するか。
46 (RSR_ALLOW_DROP_PACKAGE)	パッケージの除去を許可するか。
47 (RSR_QUERY_ISOLATION_LEVEL)	照会の実行時に使用する分離レベル
48 (RSR_ACCOUNT_STRING)	データベース・サーバーに接続中に渡す会計情報を指定するストリング
49 (RSR_ACCOUNT_OVERRIDE)	データベース・サーバーに接続中に渡すアカウント情報を指定するストリングをユーザーが変更できるか。

Value	結果が保管される long のポインター。結果は、要求されたリソース限界の値です。ブール値の場合、結果は、真の場合は非ゼロ、偽の場合はゼロとなります。 RSR_SAVE_DATA_TABLE_SPACE_NAME、RSR_DEF_COLLECTION、および RSR_ACCOUNT_STRING の場合は、-1 が戻され、戻されたストリング値について *ResourceLimit* プロパティーを問い合わせることができます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetResourceLimitEx()

short GetResourceLimitEx(short *Resource*)

説明

この関数は、要求されたリソース限界を入手します。リソース限界はサーバーごとに処理されるため、この関数を呼び出す前に、InitializeServer() を呼び出すことが必要です。この関数を呼び出した後、その結果について *ResourceLimit* プロパティーを照会してください。

パラメーター

名前	説明
Resource	リソース値は、GetResourceLimit() 呼び出しの場合と同じです。

注: 静的 SQL ステートメントには、照会 verb は使用できません。InitializeStaticQuery() から戻される照会 ID について、GetQueryVerb() は空ストリングを戻します。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetRowCount()

long GetRowCount(long *QueryID*)

説明

この関数は、現在 QMF (Windows 版) の内部バッファーに入っている行数を戻します。 QMF (Windows 版) はデータベースから受け取ったデータをバッファーに入れるため、この数は、FetchNextRow() または FetchNextRows() を用いて検索した行数よりも多い場合があります。

この関数は、データベースからすでに検索されている行数を戻します。結果セット内の合計の行数を検索したい場合には、以下の要領で検索することができます。

- CompleteQuery() を呼び出し、FetchNextRow() または FetchNextRows() を使用してすべての行を取り出します。
- Open() を呼び出すときに、FetchAllRows = TRUE を指定します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。

戻り値

正常終了の場合は、行数 (行が検索されていない場合には 0)、もしくは異常終了の場合は -1。 -1 の場合には、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetServerList()

short GetServerList(VARIANT* *List*)

説明

この関数は、QMF (Windows 版) のサーバー定義ファイル (SDF) で定義されているデータベース・サーバーの名前が入っている配列を戻します。 QMF (Windows 版) API を使用してデータベース・サーバーにアクセスしたい場合には、SDF ファイルでそのデータベース・サーバーを定義することが必要です。

パラメーター

名前	説明
List	結果が保管される VARIANT のポインター。結果は、string の配列 (変形タイプ VT_ARRAY | VT _BSTR) となり、各ストリングには 1 つのデータベース・サーバーの名前が入っています。データベース・サーバーが定義されていないと、結果は空 (変形タイプ VT_EMPTY) となります。この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。 Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

注: Microsoft Excel 7.0 および Microsoft Access 7.0 (および、Visual Basic for Applications を使用する他の 32 ビットの Microsoft プロダクトも可能性があります) におけるバグが原因で、QMF (Windows 版) から受け取った Variant 変数内のストリング・データが、ユニコード (OLE が使用) から ANSI (VBA が使用) に変換されない場合があります。この状態が起きると、ストリングの先頭文字だけが表示されます。この問題を取り除くには、その変数を使用する QMF (Windows 版) の関数を呼び出す前に、その変数を空ストリングと等しくなるように設定しておきます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。データベース・サーバーが定義されていないと、戻り値はゼロになります。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetServerListEx()

short GetServerListEx(short *Index*)

説明

この関数は、*Index* パラメーターによって参照されるサーバーの名前を検索します。この関数を呼び出した後は、戻された値について *Value* プロパティーを問い合わせることができます。

パラメーター

名前	説明
Index	サーバーのリストの索引。

戻り値

正常終了の場合はゼロ、使用可能なサーバーの数よりも索引の方が大きい場合には RS_OUTOFRANGE、異常終了の場合はゼロ以外。データベース・サーバーが定義されていなかった場合、戻り値は RS_OUTOFRANGE となります。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetStoredProcedureResultSets()

short GetStoredProcedureResultSets(long *QueryID* , VARIANT* *ResultSets*)

説明

この関数は、元の *QueryID* を用いて実行されたストアード・プロシージャーによって戻される結果セットの照会 ID を検索します。戻された各照会 ID を、FetchNextRow() または FetchNextRows() と一緒に使用すると結果セットを検索することができ、各結果セットの終わりに達したときには Close() と一緒に使用することができます。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される元の照会の ID 。

ResultSets	結果セットの照会 ID が保管される VARIANT のポインター。結果は、long integer の配列 (変形タイプ VT_ARRAY | VT _I4) となり、各長整数は対応する結果セットの照会 ID です。ストアード・プロシージャーが結果セットを戻さなかった場合、結果は空 (変形タイプ VT_EMPTY) となります。この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。 Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

GetVariables()

short GetVariables(long *QueryID* , VARIANT* *Variables*)

説明

この関数は、照会の SQL テキスト内の変数の名前の配列を戻します。 Open() または Execute() のいずれかを使用して照会を実行する前に、SetVariable() を呼び出して、その変数に値を割り当てておくことが必要です。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Variables	結果が保管される VARIANT のポインター。結果は、string の配列 (変形タイプ VT_ARRAY | VT _BSTR) となり、各ストリングには 1 つの変数の名前が入っています。 SQL ステートメントに変数がない場合、結果は空 (変形タイプ VT_EMPTY) となります。この関数を呼び出す前に、VARIANT を正しく初期化しておくことが必要です。 Visual Basic では、自動的にこれを実行します。 Visual C++ のプログラマーは、VariantInit() を呼び出すことが必要です。

注: Microsoft Excel 7.0 および Microsoft Access 7.0 (および、Visual Basic for Applications を使用する他の 32 ビットの Microsoft プロダクトも可能性があります) におけるバグが原因で、QMF (Windows 版) から受け取った Variant 変数内のストリング・データが、ユニコード (OLE が使用) から ANSI (VBA が使用) に変換されない場合があります。この状態が起きると、ストリングの先頭文字だけが表示されます。この問題を取り除くに

は、その変数を使用する QMF (Windows 版) の関数を呼び出す前に、その変数を空ストリングと等しくなるように設定しておきます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。 SQL ステートメントに変数がない場合、戻り値は RS_ERROR_NO_DATA (-1) です。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

GetVariablesEx()

short GetVariablesEx(long *QueryID* , short *Index*)

説明

この関数は、*Index* パラメーターで参照される照会の SQL テキスト内の変数の名前を戻します。この関数を呼び出した後は、戻された値について *Value* プロパティーを問い合わせることができます。 Open() または Execute() のいずれかを使用して照会を実行する前に、SetVariable() を呼び出して、この変数 (および SQL テキスト内の他のすべての変数) に値を割り当てておくことが必要です。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Variables	変数の内部リストの索引。渡された索引に対応するストリングについて、*Value* プロパティーを照会してください。 SQL ステートメントに変数がない場合、この関数は RS_ERROR_NO_DATA を戻します。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。 SQL ステートメントに変数がない場合、戻り値は RS_ERROR_NO_DATA (-1) です。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

InitializeProc()

long InitializeProc(short *SourceType* , BSTR *Source*)

説明

この関数は、プロシージャーに使用するテキストを設定します。このテキストは、この関数へのパラメーターとして渡すか、テキスト・ファイルから読み取るか、もしくは既存のプロシージャーから入手することができます。

パラメーター

名前	説明
SourceType	プロシージャー・テキストのソースを指定します。

値	意味
0 (RSS_STRING)	テキストは、*Source* パラメーターに入っています。
2 (RSS_FILE)	テキストは、*Source* パラメーターで名前が指定されたテキスト・ファイル内に入っています。
3 (RSS_QMFPROC)	テキストは、*Source* で所有者と名前が指定されたプロシージャー内に入っています。

Source	テキスト、プロシージャーの所有者と名前 (Owner.Name)、あるいはプロシージャー・ファイルが入っているファイルの名前が入っているストリング。

戻り値

正常終了の場合は、プロシージャーの ID (ProcID)。異常終了の場合は -1。*ProcID* パラメーターを必要とするすべてのインターフェース呼び出しで、この値を使用することが必要です。

InitializeQuery()

long InitializeQuery(short *SourceType* , BSTR *Source*)

説明

この関数は、照会に使用するテキストを設定します。この SQL テキストは、この関数へのパラメーターとして渡すか、テキスト・ファイルから読み取るか、もしくは既存の照会から入手することができます。照会が終了したら、close() を呼び出します。

パラメーター

名前	説明
SourceType	SQL ステートメント・テキストのソースを指定します。

値	意味
0 (RSS_STRING)	SQL テキストは、*Source* パラメーターに入っています。
1 (RSS_QMFQUERY)	SQL テキストは、*Source* で所有者と名前が指定された照会内に入っています。
2 (RSS_FILE)	SQL テキストは、*Source* パラメーターで名前が指定されたテキスト・ファイル内に入っています。

戻り値

正常終了の場合は、照会 ID。異常終了の場合は -1。 *Query* パラメーターを必要とするすべてのインターフェース呼び出しで、この値を使用することが必要です。

InitializeServer()

short InitializeServer(BSTR *ServerName* , BSTR *UserID* , BSTR *Password* , BOOL *ForceDialog* , [VARIANT *Account*], [VARIANT *SuppressDialog*])

説明

この関数は、データベース・サーバーへの接続を初期化します。 QMF (Windows 版) API で他の関数を呼び出す場合はその前に、この関数を呼び出すことが必要です。この関数は複数回呼び出すことができます。ただし、この関数を呼び出して、Commit() または Rollback() を呼び出すことで終了していないと、暗黙的にロールバックが行われます。

パラメーター

名前	説明
ServerName	使用するデータベース・サーバーの名前が入っているストリング。この名前は、QMF (Windows 版) サーバー定義ファイルに定義されている名前の 1 つと一致していなければなりません。有効なサーバーのリストを検索する場合は、GetServerList() を呼び出してください。
UserID	使用するユーザー ID が入っているストリング。ユーザー ID が NULL または空ストリングの場合、QMF (Windows 版) は、最新の照会からのユーザー ID (使用可能ならば) の使用を試みます。あるいは、QMF (Windows 版) は、ユーザー ID とパスワードを入手するための「ユーザー情報」ダイアログ・ボックスを表示します。

Password	指定されたユーザー ID についてのパスワードが入っているストリング。パスワードが NULL または空ストリングの場合、QMF (Windows 版) は、使用可能ならば (Windows for Workgroups が必要) 記憶してあったパスワードの使用を試みます。使用可能なパスワードがないと、QMF (Windows 版) は、パスワードを入手するための「ユーザー情報」ダイアログ・ボックスを表示します。
ForceDialog	ゼロ以外は、User ID と Password が指定されているかどうかに関係なく、QMF (Windows 版) が「ユーザー情報」ダイアログ・ボックスを表示することを示します。これにより、情報を使用する前に変更する機会がユーザーに提供されます。ゼロは、QMF (Windows 版) が必要な場合にのみ「ユーザー情報」ダイアログ・ボックスを表示することを示します。
Account	接続時にサーバーに渡すアカウンティング情報を指定するストリング (オプショナル)。サーバーは、ジョブ・アカウンティング・システムでこの情報を使用することができます。
SuppressDialog	ゼロ以外は、ユーザー ID とパスワードが指定されていなかった場合であっても、QMF (Windows 版) が「ユーザー情報」ダイアログ・ボックスを表示しないことを示します。このオプションは、「ユーザー情報」ダイアログ・ボックスに応答するユーザーがいない環境 (たとえば、Web サーバー) で実行するときに役立ちます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

SetParent()

InitializeStaticQuery()

long InitializeStaticQuery(BSTR *CollectionName* , BSTR *PackageName* , BSTR *ConsistencyToken* , short *SectionNumber*)

説明

この関数は、静的照会として実行したいパッケージのセクションを指定します。

パラメーター

名前	説明
CollectionName	以前にバインドされている集合の名前。
PackageName	以前にバインドされているパッケージの名前
ConsistencyToken	上記のパラメーターで命名された集合およびパッケージが使用するトークン
SectionNumber	実行する集合およびパッケージ内のステートメントのセクション番号

戻り値

正常終了の場合は、照会 ID。異常終了の場合は -1。 *QueryID* パラメーターを必要とするすべてのインターフェース呼び出しで、この値を使用することが必要です。

IsStatic()

BOOL IsStatic(long *QueryID*)

説明

この関数は、指定された照会 ID が静的照会または動的照会を示しているか否かを判別します。

パラメーター

名前	説明
QueryID	InitializeQuery() または InitializeStaticQuery() から戻される ID。

戻り値

正常終了した場合、および *QueryID* が静的照会を示している場合にはゼロ以外を戻します。それ以外の場合はゼロを戻します。

Open()

short Open(long *QueryID* , long *RowLimit* , BOOL *FetchAllRows*)

説明

この関数は、照会のためにデータベース内でカーソルを開くことによって、SELECT verb を使用する照会を実行する場合に使用します。 FetchNextRow() または FetchNextRows() を使用して、照会のためのデータを検索し、実行後は Close() を呼び出します。 QMF (Windows 版) が自動的にすべての行を取り出すように構成されている (GetResourceLimit() の説明の RSR_FETCHALLROWS

を参照) 場合、もしくは FetchAllRows パラメーターがゼロ以外の場合、QMF (Windows 版) は、この関数から戻る前に、結果セットのすべての行を取り出して内部バッファーに入れます。

注: この関数の名前は、Microsoft Access 2.0 のキーワード Open と競合します。 Microsoft Access 2.0 を使用する場合には、関数名を大括弧 [ ] で囲んでください。

注: この関数は、SQL verb の SELECT が含まれているステートメントでのみ使用してください。その他の verb (たとえば、SET) が含まれているステートメントの場合には、この関数ではなく Execute() を呼び出してください。照会が使用する verb を判別するには、GetQueryVerb() を呼び出します。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
RowLimit	データベースから検索する行の最大数を示す数値。ゼロは、QMF (Windows 版) の管理者プログラムが設定した行限界以外に強制される制限がないことを示します。
FetchAllRows	結果セット内のすべての行が自動的に取り出されて QMF (Windows 版) の内部バッファーに入れられるか否かを示すブール値。ゼロ以外の場合、すべての行が自動的に取り出され、カーソルがクローズされ、他のユーザーの使用に備えてデータベースが解放されます。これは CompleteQuery() 呼び出しの場合と同じです。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

Prepare()

short Prepare(long *QueryID*)

説明

この関数は、*QueryID* で指定された照会の準備をします。ステートメントは、データベース・サーバーによってテストされ、オブジェクトの存在、必要な許

可などが検査されます。 照会が SELECT ステートメントである場合、ステートメントによって戻される列に関する情報は、Prepare() の完了後に使用できるようになります。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

Execute()
Open()

PrintReport()

short PrintReport(long *QueryID* , short *SourceType* , BSTR *Source* , BSTR *OutputFileName* , short *PageLength* , short *PageWidth* , BOOL *IncludeDateTime* , BOOL *IncludePageNumbers* , [VARIANT *Format*], [VARIANT *UseFormPageSetup*])

説明

PrintReport() は ExportReport() の同義語です。

ReinitializeServer()

short ReinitializeServer( )

説明

この関数は、データベース・サーバーへの接続を再初期化します。通常、この関数を呼び出すことが必要になるのは、他の QMF (Windows 版) API 関数の 1 つがエラーを戻した場合だけです。この関数を呼び出すと、暗黙的にロールバックが行われ、それによって、開いたカーソルが閉じられ、すべての未解決の照会 ID は無効になります。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

Rollback()

short Rollback( )

説明

この関数は、現行の作業単位で行った変更を取り消し、現行の作業単位を終了し、開いたカーソルを閉じ、未解決のすべての照会 ID を無効にします。

注: この関数の名前は、Microsoft Access 2.0 のキーワード Rollback と競合します。 Microsoft Access 2.0 を使用する場合には、関数名を大括弧 [ ] で囲んでください。

注: ロールバックは、Open() または Execute() を呼び出すことによって実行された SQL 変更だけに影響を及ぼします。ロールバックは、その他の QMF (Windows 版) API 関数 (FastSaveData()、SaveData()、または DeleteQMFObject() など) によって行われた変更には影響を及ぼしません。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

Commit()

RunProc()

short RunProc(long *ProcID*)

説明

この関数は、指定されたプロシージャーを実行します。プロシージャーは、完了するまで、もしくはエラーが発生するまで、実行されます。このプログラミング・インターフェースを介して、プロシージャーの結果 (たとえば、実行された照会からのデータ) にアクセスすることはできません。ただし、このプロシージャーによってエクスポートされたファイルまたは保管されたデータは、実行後に使用することができます。

パラメーター

名前	説明
ProcID	InitializeProc() から戻されるプロシージャーの ID。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

SaveData()

short SaveData(long *QueryID* , long *FirstRow* , long *FirstCol* , long *LastRow* , long *LastCol* , BOOL *Replace* , BSTR *TableName* , BSTR *TableSpaceName* , BSTR *ServerName* , BSTR *UserID* , BSTR *Password* , BOOL *ForceDialog* , [VARIANT *Account*], [VARIANT *Comment*], [VARIANT *CommitScope*])

説明

この関数は、指定された範囲の行を、指定された表スペース内の指定された表に保管します。表に保管したいすべての行についての行データを検索していない場合には、この関数を呼び出す前に CompleteQuery() を呼び出すことが必要です。データベースから検索されていない行を保管しようとすると、その保管は失敗します。表がすでに存在する場合、新規のデータは、既存の表と同数および同タイプの列を持つものでなければなりません。

この関数は、他の API 関数とは別個の作業単位で実行され、その結果は自動的にコミットされます。 Commit() または Rollback() を呼び出しても、この関数を使用して行った変更には影響を及ぼしません。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
FirstRow	保管に組み込む先頭の行。結果セット内の先頭行の値は 0 です。
FirstCol	保管に組み込む先頭の列。結果セット内の先頭列の値はゼロです。
LastRow	保管に組み込む最後の行、あるいはすべての行を組み込む場合には -1。結果セット内の最後の行の値は、合計の行数より 1 小さい値です。

LastCol	保管に組み込む最後の列、あるいはすべての列を組み込む場合には -1。結果セット内の最後の列の値は、合計の列数より 1 小さい値です。
Replace	ゼロ以外は、指定されたが表の中の既存のデータと置き換わることを示します。ゼロは、指定されたデータが、表の中の既存のデータに追加されることを示します。
TableName	データを保管する表の名前。指定された表が存在しない場合には、その表が作成されます。
TableSpaceName	表が存在しているか、または表が作成される表スペースの名前。*TableSpaceName* が NULL もしくは空ストリングの場合には、デフォルトの表スペースが使用されます。常にデフォルトの表スペースを使用するように QMF (Windows 版) を構成してある (GetResourceLimit() の説明の RSR_SDDIFFERENTTS を参照) 場合には、このパラメーターは無視されます。
ServerName	表を保管するデータベース・サーバーの名前。 *ServerName* が NULL または空ストリングの場合は、InitializeServer() の呼び出しで指定したサーバーが使用され、*UserID* 、*Password* 、*ForceDialog* 、および *Account* は無視されます。
UserID	*ServerName* で別のサーバーを指定した場合、*UserID* は、そのサーバーに対して使用されるユーザー ID です。 User ID が指定されないと、QMF (Windows 版) は、このサーバーに対して指定された最後のユーザー ID (使用可能ならば) を使用するか、もしくは、使用可能なものがない場合にはダイアログ・ボックスを表示します。 *ServerName* が NULL または空ストリングの場合、このパラメーターは無視されます。
Password	*ServerName* で別のサーバーを指定した場合、*Password* は、そのサーバーに対して使用されるパスワードです。パスワードが指定されないと、QMF (Windows 版) は、このサーバーに対して指定された最後のパスワード (使用可能ならば) を使用するか、もしくは、使用可能なものがない場合にはダイアログ・ボックスを表示します。 *ServerName* が NULL または空ストリングの場合、このパラメーターは無視されます。
ForceDialog	*ServerName* で別のサーバーを指定した場合、ゼロ以外を指定すると、ユーザー ID とパスワードが指定されていた場合あるいは使用可能なものがあった場合であっても、強制的に QMF (Windows 版) に、ログオン情報の入力を求めるダイアログ・ボックスを表示させます。ゼロは、QMF (Windows 版) が必要な場合にのみこのダイアログ・ボックスを表示することを示します。 *ServerName* が NULL または空ストリングの場合、このパラメーターは無視されます。

Account	*ServerName* で別のサーバーを指定した場合、接続時にそのサーバーに渡すアカウンティング情報を指定するストリング (オプショナル)。サーバーは、ジョブ・アカウンティング・システムでこの情報を使用することができます。 *ServerName* が NULL または空ストリングの場合、このパラメーターは無視されます。
Comment	データを保管する表についてのコメントを指定するストリング (オプショナル)。
CommitScope	作業単位をコミットする前に、1 回で表に挿入する行の数 (オプショナル)。ゼロを指定すると、コミット前にすべての行を挿入することが指示されます。たとえば 10 を指定すると、10 行挿入されるたびに、コミットを実行する必要があることが指示されます。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。結果セットが空であるか、あるいはデータベースから行がまったく検索されていない場合には、FirstRow=0 および LastRow=-1 でない限り、ゼロ以外が戻されます。 FirstRow=0 および LastRow=1 の場合には、ゼロが戻され、空の表が作成されます。

SaveQMFProc()

short SaveQMFProc(BSTR *OwnerAndName* , BSTR *Text* , BSTR *Comment* , BOOL *Replace* , BOOL *Share*)

説明

この関数は、データベース・サーバーでプロシージャーを保管します。

パラメーター

名前	説明
OwnerAndName	保管するプロシージャーの所有者と名前が、ピリオドで区切られて入っているストリング。例を以下に示します。 `John.Proc2`
Text	プロシージャー内の、保管したいテキストが入っているストリング。
Comment	プロシージャーと一緒に保管したいコメントが入っているストリング。コメントがない場合には、このパラメーターを空ストリングもしくは NULL として渡します。

Replace	ゼロ以外は、同じ名前の既存のプロシージャーを置き換えます。ゼロは、同じ名前を持つ既存のプロシージャーがある場合に、操作を打ち切ります。
Share	ゼロ以外は、そのプロシージャーを他のユーザーと共用します。ゼロは、そのプロシージャーを他のユーザーと共用しません。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

SaveQMFQuery()

short SaveQMFQuery(BSTR *OwnerAndName* , BSTR *Text* , BSTR *Comment* , BOOL *Replace* , BOOL *Share*)

説明

この関数は、データベース・サーバーで照会を保管します。

パラメーター

名前	説明
OwnerAndName	保管する照会の所有者と名前が、ピリオドで区切られて入っているストリング。例を以下に示します。 `John.Query2`
Text	照会内の、保管したいテキストが入っているストリング。
Comment	照会と一緒に保管したいコメントが入っているストリング。コメントがない場合には、このパラメーターを空ストリングもしくは NULL として渡します。
Replace	ゼロ以外は、同じ名前の既存の照会を置き換えます。ゼロは、同じ名前を持つ既存の照会がある場合に、操作を打ち切ります。
Share	ゼロ以外は、その照会を他のユーザーと共用します。ゼロは、その照会を他のユーザーと共用しません。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

SetBindOption()

short SetBindOption(BSTR *CollectionName* , BSTR *PackageName* , short *Option* , short *Value*)

説明

この関数は、EndBind() を呼び出す前に、集合およびパッケージ用にオプションを設定します。

パラメーター

名前	説明
CollectionName	オプションを設定する対象のパッケージの集合 ID。
PackageName	オプションを設定する対象のパッケージの名前。
Option	下記のオプションのいずれか 1 つ。
Value	ゼロ以外は、同じ名前の既存の照会を置き換えます。ゼロは、同じ名前を持つ既存の照会がある場合に、操作を打ち切ります。
Share	指定されたオプションについての、下記の値のいずれか 1 つ。

各種のオプションの意味と値は次のとおりです。

オプション	意味	説明
DDM_PKGRPLOPT(0x211C)	同じ集合 ID と名前を持つ既存のパッケージを置き換えるか否かを指定するフラグ。	DDM_PKGRPLALW (0x241F) はい DDM_PKGRPLNA (0x2420) いいえ
DDM_STTDECDEL(0x2121)	パッケージ内の SQL ステートメントの小数点に使用する区切り文字。	DDM_DECDELPRD (0x243C) ピリオド DDM_DECDELCMA (0x243D) コンマ
DDM_STTSTRDEL(0x2120)	パッケージ内の SQL ステートメントのストリング値に使用する区切り文字。	DDM_STRDELAP (0x2426) アポストロフィ DDM_STRDELDQ (0x2427) 二重引用符

DDM_PKGISOLVL(0x2124)	パッケージの分離レベル。	DDM_ISOLVLALL (0x2443) すべて DDM_ISOLVLCHG (0x2441) 変更 DDM_ISOLVLCS (0x2442) カーソル固定 DDM_ISOLVLNC (0x2445) コミットしない DDM_ISOLVLRR (0x2444) 反復可能読み取り
DDM_PKGATHOPT(0x211E)	パッケージに関する既存の許可を保持するか否かを指定するフラグ。	DDM_PKGATHKP (0x2425) 保持する DDM_PKGATHRVK (0x2424) 取り消す
DDM_QRYBLKCTL(0x2132)	パッケージ内の、照会のためにデータの行を取り出すときに使用する方式。	DDM_FIXROWPRC (0x2418) 一度に 1 行 DDM_LMTBLKPRC (0x2417) 一度に 1 ブロック
DDM_RDBRLSOPT(0x2129)	パッケージの実行中に入手したデータベース・リソースを解放する時点。	DDM_RDBRLSCMM (0x2438) コミット時 DDM_RDBRLSCNV (0x2439) 会話の割り振り解除時
DDM_STTDATFMT(0x2122)	検索される日付値のフォーマット	DDM_ISODATFMT (0x2429) ISO DDM_USADATFMT (0x242A) US DDM_EURDATFMT (0x242B) 欧州規格 DDM_JISDATFMT (0x242C) 日本工業規格
DDM_STTTIMFMT(0x2123)	検索される時刻値のフォーマット	DDM_ISOTIMFMT (0x242E) ISO DDM_USATIMFMT (0x242F) US DDM_EURTIMFMT (0x2430) 欧州規格 DDM_JISTIMFMT (0x2431) 日本工業規格

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

SetBindOwner()

short SetBindOwner(BSTR *CollectionName* , BSTR *PackageName* , BSTR *OwnerID*)

説明

この関数は、バインディングしているパッケージについて、自分のユーザー ID とは異なる所有者を指定できるようにします。この関数は、自分のユーザー ID はパッケージをバインドするために必要な許可を持っていないが、指定された所有者はその許可を持っているという場合に必要となります。

パラメーター

名前	説明
CollectionName	所有者を指定する対象のパッケージの集合 ID。
PackageName	所有者を指定する対象のパッケージの名前。
Comment	照会と一緒に保管したいコメントが入っているストリング。コメントがない場合には、このパラメーターを空ストリングもしくは NULL として渡します。
OwnerID	バインディングしているパッケージに対して、希望する所有者 ID。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

SetBusyWindowButton()

void SetBusyWindowButton(BSTR *Text*)

説明

この関数は、使用中のウィンドウの「キャンセル」ボタンに表示されるテキストを指定します。

パラメーター

名前	説明
Text	使用中のウィンドウの「キャンセル」ボタンに表示されるテキストを指定するストリング。デフォルト値は ″Cancel″ です。空ストリングを指定すると、ボタンは隠蔽されます。指定されたテキストに関係なく、このボタンは常にウィンドウを取り消す、つまり閉じます。

戻り値
なし。

関連トピック
SetBusyWindowMessage()
SetBusyWindowMode()
SetBusyWindowTitle()
ShowBusyWindow()

SetBusyWindowMessage()

void SetBusyWindowMessage(BSTR *Message*)

説明
この関数は、使用中のウィンドウのメッセージ域に表示されるテキストを指定します。

パラメーター

名前	説明
Message	使用中のウィンドウのメッセージ域に表示されるテキストを指定するストリング。

戻り値
なし。

関連トピック
SetBusyWindowButton()
SetBusyWindowMode()
SetBusyWindowTitle()
ShowBusyWindow()

SetBusyWindowMode()

void SetBusyWindowMode(short *Mode*)

説明

この関数は、QMF (Windows 版) が使用中のウィンドウを表示するか否かを決定します。使用中のウィンドウは、ユーザーにフィードバックを提供し、ユーザーが保留中のデータベース・アクションを取り消せるようにするのに役立ちます。行った変更は、次に QMF (Windows 版) が使用中のウィンドウを表示するかまたは隠蔽することになる操作を実行したときに有効となります。

パラメーター

名前	説明
Mode	QMF (Windows 版) が使用中のウィンドウを表示する時点を指定します。

値	意味
0 (RSM_NEVER)	ウィンドウは表示されません。これがデフォルトです。
1 (RSM_WHENBUSY)	ウィンドウは、QMF (Windows 版) がデータベースと通信していて使用中のときに表示されます。 QMF (Windows 版) は、適宜、自動的にこのウィンドウを表示します。
2 (RSM_CLIENTCONTROLLED)	ウィンドウは、ShowBusyWindow(TRUE) を呼び出した後、および ShowBusyWindow(FALSE) を呼び出した後に表示されます。クライアントが、ウィンドウを表示する時点を決定します。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

SetBusyWindowButton()
SetBusyWindowMessage()
SetBusyWindowTitle()
SetParent()
ShowBusyWindow()

SetBusyWindowTitle()

void SetBusyWindowTitle(BSTR *Title*)

説明
この関数は、使用中のウィンドウのタイトル・バーに表示されるテキストを指定します。

パラメーター

名前	説明
Title	使用中のウィンドウのタイトル・バーに表示されるテキストを指定するストリング。

戻り値
なし。

関連トピック
SetBusyWindowButton()
SetBusyWindowMode()
SetBusyWindowMessage()
ShowBusyWindow()

SetGlobalVariable()

short SetGlobalVariable(BSTR *Name* , BSTR *Value*)

説明
この関数は、指定されたグローバル変数に値を割り当てます。この値は、照会、書式、およびプロシージャーで使用することができます。

パラメーター

名前	説明
Name	設定する変数の名前が入っているストリング。
Value	指定された変数に割り当てたい値が入っているストリング。

戻り値
正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

SetHostVariable()

short SetHostVariable(long *QueryID* , VARIANT *Index* , VARIANT *Value*)

説明

この関数は、照会によって参照される、指定されたホスト変数に値を割り当てます。照会は、ホスト変数 (QMF 照会を用いて保管されたか、または AddHostVariable() によって作成されたもの) を参照する静的照会でなければなりません。 *Index* では、ホスト変数の数値索引、またはホスト変数の名前のいずれをも指定することができます。

パラメーター

名前	説明
QueryID	InitializeStaticQuery() から戻される照会の ID。
Index	照会でホスト変数の索引を指定する number (変形タイプ VT_I2) 、もしくは、ホスト変数の名前を指定する string (変形タイプ VT_BSTR) のいずれか。
Value	ホスト変数の値。ヌル値を指定する場合は、変形のタイプを VT_EMPTY に設定する必要があります。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

SetOption()

short SetOption(short *Mode,* VARIANT *Value*)

説明

この関数は、QMF (Windows 版) で指定されたオプション値を設定します。オプションの中には、QMF (Windows 版) が再始動されるまでは変更が有効にならないものもあります。通常の状態では、QMF (Windows 版) API オブジェクトのすべてのインスタンスを破棄してしまうまで、QMF (Windows 版) の再始動は行われません。

パラメーター

名前	説明
Option	設定するオプションを指定します。

値	意味
0 (RSO_SERVER_DEFINITION_FILE)	サーバーの定義ファイル名。
1 (RSO_CPIC_DLL)	CPI-C プロバイダーの DLL ファイル名。

2 (RSO_CPIC_TIMEOUT_WARNING)	CPI-C 警告タイムアウト (秒数)。この制限は、QMF (Windows 版) API には使用されません。
3 (RSO_CPIC_TIMEOUT_CANCEL)	CPI-C 取り消しタイムアウト (秒数)。
4 (RSO_TCP_TIMEOUT_WARNING)	TCP 警告タイムアウト (秒数)。この制限は、QMF (Windows 版) API には使用されません。
5 (RSO_TCP_TIMEOUT_CANCEL)	TCP 取り消しタイムアウト (秒数)。
6 (RSO_DISPLAY_NULLS_STRING)	ヌル値を表示するために使用するストリング。
7 (RSO_ENTER_NULLS_STRING)	ヌル値を入力するために使用するストリング。
8 (RSO_ENTER_DEFAULTS_STRING)	デフォルト値を入力するために使用するストリング。
9 (RSO_TRACE_FILE_1)	トレース・ファイル 1 の名前。
10 (RSO_TRACE_FILE_2)	トレース・ファイル 2 の名前。
11 (RSO_TCP_TRACE_LEVEL)	TCP トレース・レベル。
12 (RSO_CPIC_TRACE_LEVEL)	CPI-C トレース・レベル。
13 (RSO_DDM_TRACE_LEVEL)	DDM トレース・レベル。

名前	説明
Value	オプションに設定する値。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

GetOption()

SetParent()

short SetParent(long *ParentWnd*)

説明

この関数は、ダイアログ用の親ウィンドウを設定します。通常、QMF (Windows 版) がダイアログを表示する (使用中のウィンドウ、もしくは「ユーザー情報」ダイアログ・ボックス内に) 場合、 そのダイアログが中心に置かれ、QMF (Windows 版) のメイン・ウィンドウの様式にならいます。この関数

は、QMF (Windows 版) ダイアログ・ボックスを強制的に中心に置き、クライアント・アプリケーション・ウィンドウの形式にならうことができるようにします。

パラメーター

名前	説明
ParentWnd	新規の親ウィンドウの HWND。親として QMF (Windows 版) のメイン・ウィンドウを使用する場合は、NULL を指定してください。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック

ShowBusyWindow()

SetProcVariable()

short SetProcVariable(long *ProcID* , BSTR *Name* , BSTR *Value*)

説明

この関数は、指定された変数に値を割り当てます。この値は、プロシージャーを実行する前に、変数との置換が行われます。プロシージャー内に複数の変数がある場合は、RunProc() を呼び出す前に、この関数を呼び出して、変数の値を設定しておくことが必要です。

パラメーター

名前	説明
ProcID	InitializeProc() から戻されるプロシージャーの ID。
Name	設定したい変数の名前が入っているストリング。
Value	指定された変数に割り当てたい値が入っているストリング。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

SetVariable()

short SetVariable(long *QueryID* , BSTR *Name* , BSTR *Value*)

説明

この関数は、指定された変数に値を割り当てます。この値は、SQL ステートメントを実行する前に、変数との置換が行われます。 SQL ステートメント内に複数の変数がある場合は、Open() または Execute() を呼び出す前に、この関数を呼び出して、変数の値を設定しておくことが必要です。

パラメーター

名前	説明
QueryID	InitializeQuery() から戻される照会の ID。
Name	設定したい変数の名前が入っているストリング。
Value	指定された変数に割り当てたい値が入っているストリング。

戻り値

正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType() を呼び出すと、詳しいエラー情報を入手することができます。

ShowBusyWindow()

void ShowBusyWindow(BOOL *Show*)

説明

この関数は、QMF (Windows 版) に、使用中のウィンドウを表示または隠蔽するように通知します。使用中のウィンドウは、ユーザーにフィードバックを提供し、ユーザーが保留中のデータベース・アクションを取り消せるようにするのに役立ちます。この関数は、RSM_CLIENTCONTROLLED モードを指定して SetBusyWindowMode() を呼び出した場合にのみ作動します。 SetParent() を呼び出して親ウィンドウを設定した場合、使用中のウィンドウは指定されたウィンドウにならいます。

パラメーター

名前	説明
Show	ゼロ以外は使用中のウィンドウを表示し、ゼロは使用中のウィンドウを隠蔽します。ゼロ以外の場合、使用中のウィンドウは、*Show* をゼロに設定して、ShowBusyWindow() を呼び出すまで表示されます。

戻り値
なし。

StartBind()

short StartBind(BSTR *CollectionName* , BSTR *PackageName* , BSTR *ConsistencyToken*)

説明
この関数は、データベース内のパッケージのバインディング処理を開始します。

パラメーター

名前	説明
CollectionName	パッケージに対して、希望する集合 ID。
PackageName	パッケージに対して、希望する名前。
ConsistencyToken	データベース内のバインド済みパッケージと、そのパッケージを使用するアプリケーションとの間の整合性を保証するために使用される、8 バイト・トークンの 16 進表記が入っている 16 文字の長さのストリング。パッケージ内のセクションを実行するとき、この同じ値を提供する必要があります。

戻り値
正常終了の場合はゼロ、異常終了の場合はゼロ以外。戻り値がゼロ以外である場合は、GetLastErrorString() または GetLastErrorType()、GetLastSQLCode()、GetLastSQLError()、GetLastSQLState() を呼び出すと、詳しいエラー情報を入手することができます。

関連トピック
EndBind()
CancelBind()

付録. 特記事項

本書において、日本では発表されていない IBM 製品 (機械およびプログラム)、プログラミングまたはサービスについて言及または説明する場合があります。しかし、このことは、弊社がこのような IBM 製品、プログラミングまたはサービスを、日本で発表する意図があることを必ずしも示すものではありません。本書でIBM ライセンス・プログラムまたは他の IBM 製品に言及している部分があっても、このことは当該プログラムまたは製品のみが使用可能であることを意味するものではありません。 IBM 製品、プログラム、またはサービスに代えて、 IBM の有効な知的所有権またはその他の法的に保護された権利を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM によって明示的に指定されたものを除き、他社の製品と組み合わせた場合の操作の評価と検証はお客様の責任で行っていただきます。

IBM は、本書で解説されている主題について特許権 (特許出願を含む)、商標権、または著作権を所有している場合があります。本書の提供は、これらの特許権、商標権、および著作権について、本書で明示されている場合を除き、実施権、使用権等を許諾することを意味するものではありません。実施権、使用権等の許諾については、下記の宛先に書面にてご照会ください。

〒106-0032 東京都港区六本木 3 丁目 2-31
AP 事業所
IBM World Trade Asia Corporation
Intellectual Property Law & Licensing

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

本書に対して、周期的に変更が行われ、これらの変更は、文書の次版に組み込まれます。 IBM は、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム（本プログラムを含む）との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
J74/G4
555 Bailey Avenue
P.O Box 49023
San Jose, CA 95161-9023
U.S.A

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストはおこなっておりません。また、IBM 以外の製品に関するパフォーマンスの正確性、互換性、またはその他の要求は確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書の情報は計画の目的でのみ使用されるものです。これらの情報は、記載された製品が使用可能になる前に変更される場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権表示

著作権使用許諾:

本書には、OS/2 でのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。サンプル・ソース・コードのすべての部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。「(C) (お客様の会社名) (西暦年). All rights reserved.」

この情報をソフトコピーで見ている場合には、写真やカラー・イラストが表示されない場合があります。

商標

以下の用語は、IBM Corporation の米国ならびに他の国における商標です。

ACF/VTAM
Advanced Peer-to-Peer Networking
AIX
AIX/6000
AS/400
CICS
CICS/ESA
CICS/MVS
CICS/VSE
COBOL/370
DATABASE 2
DataJoiner
DB2
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
DXT
GDDM
IBM
IBMLink
IMS
Language Environment
MVS/ESA
MVS/XA
OfficeVision/VM
OS/2
OS/390
PL/I
PROFS
QMF
RACF
S/390
SQL/DS
Virtual Machine/Enterprise Systems Architecture
Visual Basic
VM/XA
VM/ESA
VSE/ESA
VTAM

Java またはすべての Java ベースの商標とロゴ、および Solaris は、米国ならびに他の国における Sun Microsystems,Inc の商標です。

Lotus および 1-2-3 は、米国ならびに他の国における Lotus Development Corporation の商標です。

Microsoft、Windows、および Windows NT は、Microsoft Corporation の登録商標です。

2 つのアスタリスク (**) で示されている、その他の会社名、製品名、およびサービス名は、他社の商標または登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

[サ行]

[タ行]

[ナ行]

[ハ行]

R

S

ファイル番号:
プログラム番号: 5675-DB2
5697-F42
5697-G24
5697-G22
5648-D35
5697-G23

Printed in Japan

SC88-8670-00

日本アイ・ビー・エム株式会社
〒106-8711　東京都港区六本木3-2-12

Spine information:

QMF

QMF (Windows 版) V7 入門

バージョン 7